VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EVOLUČNÍ NÁVRH SIMULÁTORU ZALOŽENÉHO NA CELULÁRNÍCH AUTOMATECH

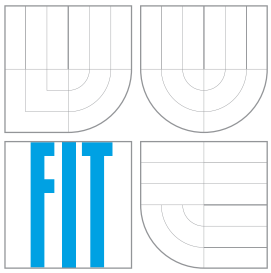
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VLADIMÍR BRIGANT

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EVOLUČNÍ NÁVRH SIMULÁTORU ZALOŽENÉHO NA CELULÁRNÍCH AUTOMATECH

EVOLUTIONARY DESIGN OF SIMULATOR BASED ON CELLULAR AUTOMATA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — VLADIMÍR BRIGANT
AUTHOR

VEDOUCÍ PRÁCE — Ing. MICHAL MRNUŠTÍK
SUPERVISOR

BRNO 2011

Abstrakt

Tato práce popisuje návrh simulátoru založeného na celulárních automatech, který je schopen předpovědet chování komplexního prostorového systému. Tato predikce je založena na dostupních datech a přechodové funkci získané pomocí regresní analýzy ve spolupráci s evolučními algoritmy. Dvě metody regresné analýzy (lineární a logistická regrese) jsou navrhnuty, implementovány a porovnány na predikci rastu urbanizace města Brno.

Abstract

This work describes concept of a cellular automata (CA) simulator, which is able to predict behaviour of a complex spatial system. This prediction is based on available training data and transition rule acquired from regression analysis powered by evolutionary algorithms. Two regression analysis methods (linear and logistic regression) are suggested, implemented and compared on urban growth prediction of Brno city.

Klíčová slova

simulátor, celulární automat, evoluční návrh, genetické algoritmy, lineární regrese, logistická regrese, urbanizace, Brno

Keywords

simulator, cellular automaton, evolutionary design, genetic algorithms, linear regression, logistic regression, urban land-use, Brno

Citace

Vladimír Brigant: Evoluční návrh simulátoru založeného
na celulárních automatech, bakalářská práce, Brno, FIT VUT v Brně, 2011

Evoluční návrh simulátoru založeného na celulárních automatech

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Mrnuštíka

.....
Vladimír Brigant
16. května 2011

Poděkování

Chcel by som poďakovať Ing. Michalovi Mrnuštíkovi za cenné rady pri tvorbe tejto práce.

Obsah

Zoznam obrázkov

Kapitola 1

Úvod

Grom tejto práce je simulácia. Simulácia všeobecne je napodobňovanie nejakého chovania, príkladom môže byť simulácia nočnej oblohy. Existuje veľké množstvo programov, ktoré slúžia na simuláciu nočnej oblohy nielen Zeme, ale aj iných planét a mesiacov, dokonca aj hviezd. V prípade hviezd sa však už nedá hovoriť o nočnej oblohe. Modelovaným systémom je v tomto prípade viditeľná časť vesmíru, modelovým systémom je počítačový program. S takýmto simulátorom môžeme experimentovať napríklad tak, že zrýchlime plynutie času a budeme schopný pozorovať rýchle striedanie dňa a noci, rýchly pohyb hviezd a celých galaxii. A práve toto experimentovanie je príčinou tvorby modelu, na základe experimentov sa pokúšame získať nové vedomosti o skúmanom systéme a tie zase prípadne použiť pri korekcii modelu. Teda môžeme povedať, že simulácia je napodobňovanie nejakého systému pomocou iného systému (napríklad počítačového programu) za účelom získať nové informácie o skúmanom systéme pomocou experimentov.

Dnes si už len máloktorý automobilový koncern vie predstaviť testovanie svojich bezpečnostných princípov poškodzovaním x automobilov pri každej sade testov. Výsledky by boli presné a vierohodné, ale takýto spôsob testovania by bol krajne neefektívny, pretože v dnešnej dobe, dobe pomerne rýchlych procesorov a objemných pamätí, sa niekoľkonásobne oplatí vytvoriť počítačový model, napríklad automobilu, a s týmto modelom pracovať, experimentovať.

Častým kameňom úrazu je však tvorba vlastného modelu. V rámci modelu nie sme schopný obsiahnuť úplne všetko, ale snažíme sa doň zahrnúť všetko podstatné, čo by mohlo mať vplyv na informácie, ktoré sa budeme snažiť experimentovaním získať. Všetko ostatné abstrahujeme, zanedbáme. Pri simulácií čelného nárazu s kamiónom nemá zmysel modelovať farbu laku automobilu a aký odtieň bude mať určitá časť plechu pri definovanom osvetlení. Pravdepodobne by bolo v silách návrhárov a analytikov vytvoriť model, ktorý by sa blížil reálnemu systému, ale bolo by to časovo a prostriedkovo tak náročné, že by sa viac oplatilo rozbíjanie automobilov a kamiónov.

Problém tvorby modelu nejakého reálneho systému je možné uchopiť rôznymi spôsobmi. Systémy, ktorých chovanie sa budeme snažiť predpovedať, sú priestorovo orientované a ich chovanie je veľmi komplikované. V tejto práci budem prezentovať evolučný návrh simulátoru, ktorý bude založený na veľmi jednoduchom a v súčasnosti rozšírenom modeli popísanom v kapitole 2, na modeli celulárneho automatu (CA). Najväčšími zbraňami CA sú jednoduchosť a lokalita. Mozgom celulárnych automatov je ich prechodová funkcia, ktorá môže mať rôzne podoby. Existuje klasický typ prechodovej funkcie, ktorá závisí len na stavoch okolitých buniek. Na praktické využitie však častokrát nepostačuje.

Celulárny automat v našom návrhu bude využívať o niečo sofistikovanejšiu funkciu za-

bezpečujúcu evolúciu CA, ktorá bude počítať nasledujúce stavy buniek CA z určitých vlastností, ktoré pre danú bunku platia. Je vhodné podotknúť, že náš simulátor bude pracovať len s konkrétnymi konfiguráciami CA a definovanými vlastnosťami jeho buniek. Vhodnú prechodovú funkciu sa budeme snažiť získať z týchto dát (konfigurácii CA), ktoré máme k dispozícii. Analýzou dát sa dostávame do štatistiky, kde existujú rôzne modely na hľadanie vzťahov medzi dátami. My použijeme regresnú analýzu, konkrétne lineárnu a logistickú regresiu. Logistická regresia sa využíva hlavne na odhad kategórie závislej premennej, ktorej hodnoty nie je možné nejakým spôsobom usporiadať, na základe nezávislých premenných. Nezávislé premenné budú pre nás vlastnosti resp. parametre buniek CA a závislou premennou bude stav bunky. V návrhu bude vystupovať aj spomínaná lineárna regresia, ktorá je vhodná v prípade lineárnych vzťahov. Bližší popis týchto regresných techník sa nachádza v kapitole 4. Jedným z cieľov je ich porovnanie na zvolenom probléme.

Väčšina systémov, ktoré sa v súčasnosti pokúšame skúmať, je veľmi zložitá a komplexná. Samozrejme, dá sa chovanie systému v danom čase a prostredí definovať pomocou matematického aparátu, väčšinou ide o diferenčné, diferenciálne, či parciálne diferenciálne rovnice, ale najmä ich zostavenie nebýva jednoduché a častokrát obsahujú veľký počet parametrov, takže aj výpočet býva veľmi časovo náročný.

Tým sa dostávame k otázke, prečo bude návrh simulátoru evolučný? V rámci logistickej a lineárnej regresie potrebujeme zistiť, ako veľmi vplýva daná vlastnosť bunky na jej nasledujúci stav. Teda budeme potrebovať nejakým spôsobom zistiť váhy jednotlivých vlastností. Tu prichádza chvíľa jedného z najväčších prírodovedcov 19. storočia. Darwinova teória, aj keď už bez jeho autora, s nami žije už nejaký ten piatok a jej vplyv na vedu bol, je a aj bude značný. Princípy prirodzeného výberu, teda selekcie, ale aj ďalšie princípy, ktoré sa v slávnom Darwinovom diele nenachádzali, ako je napríklad mutácia, sa veľmi dobre adaptovali v informačných kruhoch. Evolučný návrh z pohľadu informatiky znamená aplikáciu evolučných algoritmov na riešený problém. Výhodou evolučného návrhu je pomerná jednoduchosť tvorby modelu, nevýhodou je však to, čo môžeme pozorovať aj v klasickej biologickej evolúcii – vysoká „časová zložitosť“, ktorá je však častokrát omnoho výhodnejšia ako riešiť problém inými postupmi.

Evolučné algoritmy(EA) sa zaraďujú medzi optimalizačné metódy, ktoré poskytujú čo možno najlepšie riešenie za rozumný čas. Výsledkom väčšinou nebýva optimálne, ale len suboptimálne riešenie, s ktorým sa v mnohých prípadoch uspokojíme. EA sa začali používať už v 60–tych rokoch 20. storočia. V tej dobe však neboli počítače technologicky na dnešnej úrovni, výraznému rozmachu sa dožili v poslednom desaťročí. Postupom času sa vytvorili 4 skupiny evolučných algoritmov. Bližší popis je obsiahnutý v tejto práci, konkrétne v kapitole 3, na úvod len spomeniem, že v evolučnom návrhu pri tzv. korelácii váh vlastností buniek CA budem využívať evolučné stratégie, ktoré sú až na pár rozdielov totožné s genetickými algoritmami inšpirovanými genetikou. Evolučnému návrhu simulátoru sa venuje kapitola 4, implementačnej stránke návrhu kapitola 5.

K čomu by náš simulátor mal slúžiť? Mal by byť schopný predpovedať chovanie priestorových dynamických systémov, ktoré vykazujú komplexné chovanie závisiace na veľkom množstve rôznych parametrov. Asi najväčšou aplikačnou doménou je predikcia rastu urbanizácie miest alebo obcí. V kapitole 6 je riešený problém rastu urbanizácie mesta Brna. Sú použité obe spomenuté metódy regresnej analýzy. Zhodnotené sú výsledky, klady a zápory. Simulátory tohto typu sa dajú použiť aj v iných oblastiach, všade tam, kde ide o priestorové, dynamické a komplexné javy. Príkladom môžu byť rôzne geologické, sociologické či biologické procesy.

Kapitola 2

Koncept celulárneho automatu

Ako som načrtol v úvodnej kapitole, model celulárneho automatu (CA) je veľmi jednoduchý a pomocou jednoduchých pravidiel sme schopný dosiahnuť komplexného chovania. V tejto kapitole bude tento model viac priblížený, načrtnuté historické pozadie a súvislosti.

2.1 Historické pozadie

Koncept celulárneho automatu má za sebou dlhú púť. John von Neumann uviedol na prednáške s názvom „Všeobecná teória automatov“ svoje myšlienky ohľadom takého výpočetného systému, ktorý by bol schopný vlastnej reprodukcie. Inšpiroval sa prácami vedcov z oblasti umelých neurónových sietí – W. McCullocha a W. Pittsa. Neumann navrhol samoreprodukujúci sa automat a pritom došiel k záveru, že na svojej najnižšej úrovni vedie komplexita k degenerácii. To znamená, že automat, ktorý tvorí iné automaty, ich tvorí v jednoduchšej forme než je on sám. Ale nad určitou minimálnou úrovňou toto prestáva platiť a automat dokáže udžovať sám seba. Neumann dokázal spojiť matematickú teóriu, ktorá sa skrýva za automatmi a biológiu. Vytvoril automat s 29 stavmi, 200 000 bunkami, pričom neuróny zaisťovali logické riadenie automatu[14].

Jeho riešenie však bolo komplikované a spolu so Stanislawom Ulamom vytvorili oveľa jednoduchšie a elegantnejšie riešenie – celulárny automat. Týmto položili základy pre štúdium tohto konceptu. Po Neumannovej smrti pokračovali v štúdiu celulárnych automatov A. Burks a jeho študent J. Holland, ktorý je však známejší z inej oblasti, z oblasti genetických algoritmov.

Vtedajšie výkonnostne slabé výpočetné zariadenia a skeptizmus k novému prístupu spôsobili znižovanie záujmu o celulárne automaty. Oživenie do tejto sféry priniesol v roku 1970 matematik J. H. Conway, ktorý vytvoril hru „Life“ založenú na celulárnych automatoch. Krása tejto hry spočíva v tom, že vďaka veľmi jednoduchému pravidlu je schopná vykazovať komplexné chovanie.

Vďaka paralelizmu celulárnych automatov začali vznikať ich hardwarové implementácie. Azda najznámejšou je CAM (Cellular Automata Machine) v niekoľkých verziách vyrobených na MIT (Massachusetts Institute of Technology), ktorej vývoj majú na svedomí N. Margolus a T. Toffoli.

Za jedno z najlepších a najrozsiahlejších diel o celulárnych automatoch sa považuje kniha Stephena Wolframa z roku 2002, ktorý dokonca hovorí o novom druhu vedy [35]. Terry Sejnowski, odborník na neuronové siete zo Salkovho Inštitútu pre biologické štúdie, o Wolframovi napísal [17]:

„Je nesporne jedným z najinteligentnejších vedcov našej planéty. Jeho kniha je fascinujúca a bude mať rovnaký dopad ako Newtonova Principia.“

Wolfram hovorí, že súčasná veda by sa na veci mala začať dívať z iného uhla, samozrejme v rámci možností. Všetky komplexné systémy, ktoré dokážeme popísať, väčšinou popisujeme tak, že systém rozložíme na najmenšie elementy a skúmaním týchto častíc sa snažíme popísať systém ako celok. Dokážeme popísať chovanie častíc, ale nedokážeme popísať, akým spôsobom tieto častice spolupracujú, akým spôsobom dokážu vytvoriť komplexný systém schopný komplexného chovania, komplexného z nášho pohľadu. Príkladom môže byť akýkoľvek organizmus, napríklad aj naše ľudské telo. V súčasnosti sme schopní popísať telo ako celok, vieme ako sa chová cievny systém, nervový systém a druhým extrémom je, že vieme popísat tie najmenšie častice, z ktorých sa dané subsystémy skladajú. Tieto najmenšie častice sú samozrejme najmenšie z pohľadu úrovne nášho poznania, v dobe písania tejto práce sú najmenšími časticami v terminológii kvantovej fyziky leptóny(elektrón, neutríno) a kvarky, z ktorých sú zložené napríklad protóny. Ale to, ako, čo a prečo ľudské telo funguje, zatiaľ presne známe nie je. Ľudské telo je v podstate len zoskupenie týchto „najmenších“ častíc, ktoré vykazujú emergentné chovanie hodné obdivu.

Ďalším príkladom môže byť snaha o vysvetlenie chovania vesmíru. Einsteinova klasická a všeobecná teória relativity popisuje sily gravitácie a veľkorozmernú štruktúru vesmíru, tj. štruktúry na škále od niekoľkých kilometrov až po 10^{24} kilometrov – čo je rozmer pozorovatelnej časti vesmíru. Kvantová teória sa naopak zaoberá a dokonale popisuje javy v nesmierne malých merítkach okolo 10^{-12} metra[13]. Veci sa snaží dať do poriadku teória superstrún alebo M – teória, ktorej pravdivosť však nie je možné v dohľadnej dobe experimentálne dokázať ani vyvrátiť, takže sa dá povedať, že hraničí s filozofiou[11].

Teda podľa Wolframa komplexné chovanie je výsledkom v princípe jednoduchých systémov, ktoré sa riadia jednoduchými pravidlami, čo je prípad práve celulárnych automatov.

2.2 Model CA

Celulárny automat je dynamický systém a matematický model, ktorý má diskrétny čas aj priestor. Model celulárneho automatu môže byť všeobecne n-dimenzionálny, v praxi sa však väčšinou stretávame s 1D, 2D alebo 3D celulárnymi automatmi.

Princíp obyčajného celulárneho automatu je jednoduchý. Máme určitý počet buniek, ktoré sú usporiadané do istej topológie, väčšinou sa využíva klasická pravouhlá topológia. Každá bunka sa môže nachádzať v jednom z možných stavov. Stav, ktorý bunka nadobudne v ďalšom časovom úseku je daný na základe prechodovej funkcie, ktorá je rovnaká pre všetky bunky. Podľa [10], formálny zápis modelu CA by mohol vyzerať nasledovne:

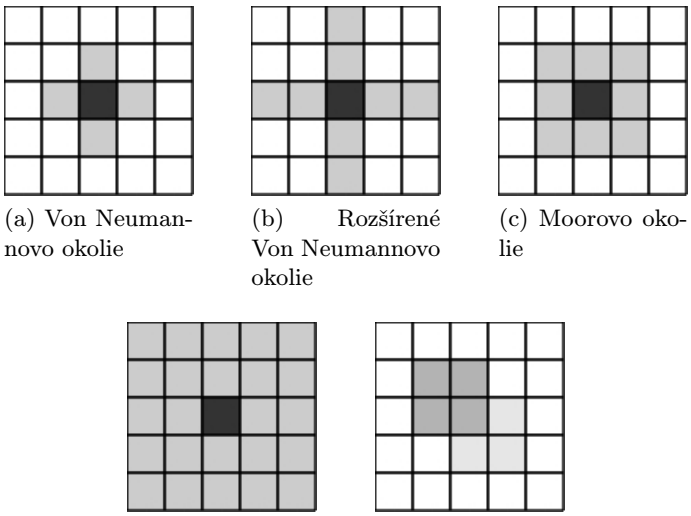
$$s^{(t+1)} = f(s^{(t)}, s_N^{(t)}), \forall i, j, \tag{2.1}$$

kde $s^{(t+1)}$ reprezentuje stav bunky danej pozíciami i a j v čase $t+1$, $s^{(t)}$ znamená stav bunky v čase t, f je prechodová funkcia CA a $s_N^{(t)}$ reprezentuje stavy okolitých buniek.

Okolie môže byť rôzne, základné typy okolia je možné vidieť na obrázku 2.1. Bližšie popíšem Margolusove okolie znázornené na obrázku 2.1e, ktoré nie je až tak intuitívne ako ostatné. Tento typ okolia je špecifický, pretože plocha s bunkami sa rozdelí na štvorce veľkosti 2x2 a stav všetkých buniek v štvorci závisí len na 4 bunkách ohraničených daným

štvorcom. Navyše, všetky bunky v jednotlivých štvorcových plochách majú rovnaký stav. Aby sa však nebránilo propagácii stavov buniek, celá sieť 2x2 plôch sa posunie v každom párnom kroku evolúcie automatu o jednu bunku v ose x aj y a v každom nepárnom kroku zase späť. Tento pomerne zvláštny typ okolia sa úspešne využíva napríklad pri približnom riešení Navier–Stokesových rovníc [31].

Celulárne automaty sa delia podľa uniformity na 2 skupiny: uniformné a neuniformné. Uniformné CA sú charakteristické tým, že pre každú bunku platí rovnaká prechodová funkcia. Naopak neuniformné môžu obsahovať bunky, ktoré majú odlišnú prechodovú funkciu.



(a) Von Neumannovo okolie

(b) Rozšírené Von Neumannovo okolie

(c) Moorovo okolie

(d) Rozšírené Moorovo okolie

(e) Margolusovo okolie

Obrázok 2.1: Rôzne typy okolia CA

Hlavné rysy celulárneho automatu sú:

- **paralelizácia** – jednotlivé stavy buniek je možné počítať paralelne, existujú aj rôzne harwarové implementácie šité na mieru, medzi najznámejšie patrí už spomínaná CAM (Cellular Automata Machine), v dobe písania tejto práce bola najaktuálnejšia verzia CAM-8[1]
- **lokalita** – nový stav bunky závisí na aktuálnom stave bunky a stavoch okolitých buniek
- **homogenita** – všetky bunky používajú rovnakú prechodovú funkciu, toto však platí len pre klasické celulárne automaty (uniformné)
- **diskrétnosť** – časová aj priestorová

Výhody celulárneho automatu:

- **jednoduchosť** – či už ide o jednoduchosť principiálnu alebo implementačnú
- **paralelizmus** – stavy buniek v nasledujúsom časovom úseku je možné počítať paralelne, čo predstavuje obrovskú výhodu a rýchlostný potenciál evolúcie CA

[1]Pre bližšie informácie o CAM-8 navštívte http://www.ai.mit.edu/projects/im/cam8/

- **vizuálnosť** – výstupy sú väčšinou vizualizovatelné a vhodné na spracovanie príslušnými nástrojmi, napríklad geografickými informačnými systémami

Samozrejme, všetko má svoje klady aj zápory, medzi nevýhody CA patrí:

- Pamäťová náročnosť modelu rapídne stúpa pri nepatrnom zväčšení okolia či zvýšení počtu stavov, ktorých môžu bunky nadobúdať. Porovnanie náročnosti modelu v závislosti na počte stavov a type resp. veľkosti okolia je znázornené na obrázku 2.2.
- Časová náročnosť modelu tiež rapídne stúpa pri nepatrnom zväčšení okolia alebo počtu stavov v prípade kvaziparalelného výpočtu.
- Diskrétnosť sa v niektorých prípadoch nehodí a môže byť kontraproduktívna.
- Klasický CA má rôzne obmedzenia ako napríklad rovnaký typ okolia pre všetky bunky, tieto nedostatky riešia rôzne rozšírenia klasického konceptu CA.

Počet stavov	2	5	7	s
Počet konfigurácii	2^n	5^n	7^n	s^n
Dátová štrukt. pravidla	bitové pole	„lookup“ table	„lookup“ table	„lookup“ table
Veľkosť pravidla	2^{n-3} B	5^n B	7^n B	s^n B
1D 3 okolie	1 B	125 B	343 B	s^3 B
1D 5 okolie	4 B	3 kB	16,4 kB	s^5 B
2D Neumann	4 B	3 kB	16,4 kB	s^5 B
2D Moore	64 B	1,86 MB	38,48 MB	s^9 B
2D Ext. Neumann	64 B	1,86 MB	38,48 MB	s^9 B
2D Ext. Moore	4 MB	264,7 PB[1]	1,14 ZB[2]	s^{25} B
3D Neumann	16 B	76,3 kB	804,2 kB	s^7 B
3D Moore	16 MB	6,46 EB[3]	55,66 ZB	s^{27} B
3D Ext. Neumann	1 kB	1,114 GB	90,23 GB	s^{13} B
3D Ext. Moore	$5,32 \cdot 10^{36}$ B	$1,94 \cdot 10^{63}$ YB[4]	$3,59 \cdot 10^{81}$ YB	s^{125} B

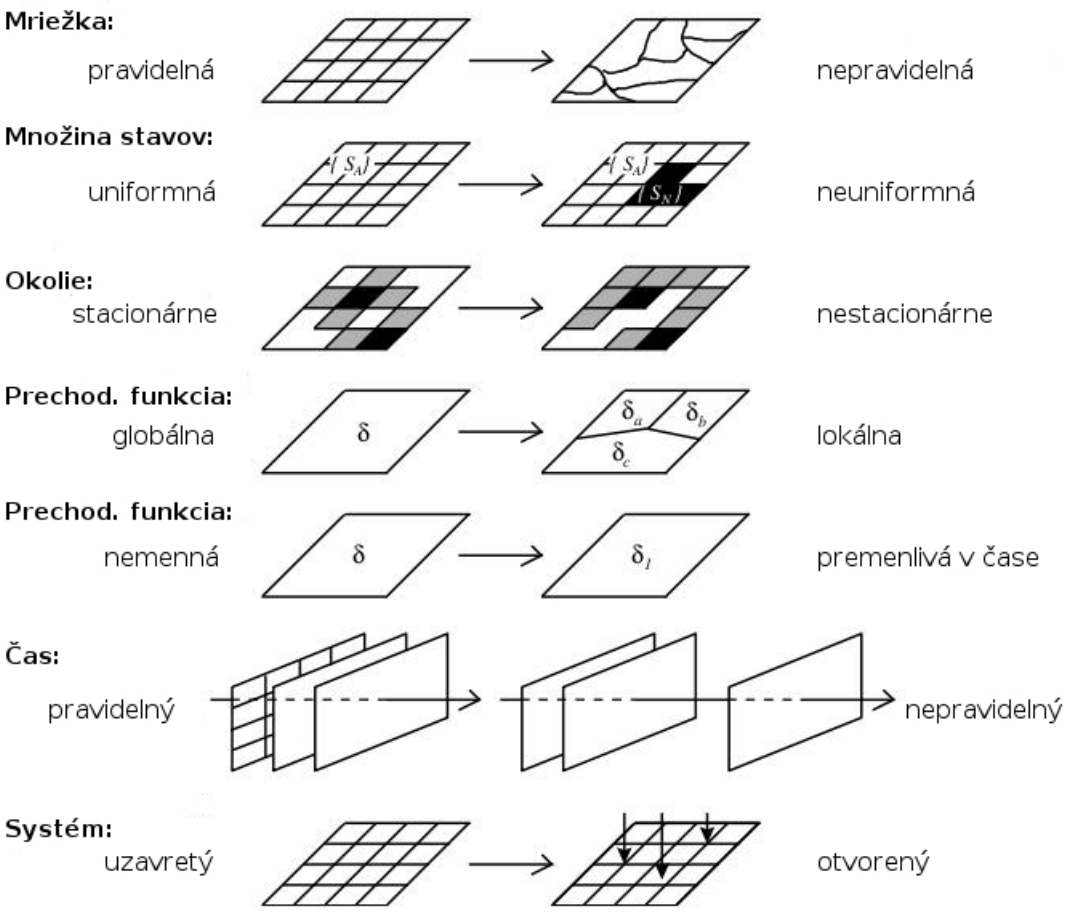
Obrázok 2.2: Pamäťová náročnosť modelu celulárneho automatu v závislosti na počte stavov a type okolia

2.3 Aplikácie modelu CA

Pomerne úzka aplikačná doména celulárnych automatov sa postupom času rozrástla až do takej miery, že sa CA začali používať v každej oblasti, kde je potrebné simulovať nejaké dynamické priestorové deje, ktoré sú charakteristické svojou komplexnosťou. V praxi sa CA využívajú v rôznych modifikáciach, málokedy sa použije model klasického celulárneho automatu. Príklady modifikácii CA sú znázornené na obrázku 2.3. Veľké uplatnenie nachádzajú celulárne automaty vo fyzikálnych dejoch [6], biologických procesoch, geologických procesoch, ekonomických predpovediach atď.

[1] PB = petabyte = 10^{15} B
[2] ZB = zettabyte = 10^{21} B
[3] EB = exabyte = 10^{18} B
[4] YB = yottabyte = 10^{24} B

Celulárne automaty je možné použiť a aj sa používajú na tzv. výpočetné problémy ako je napríklad problém majority alebo problém synchronizácie. Vedci zo Santa Fe Institute sa zaoberali týmito problémami s tým, že prechodová funkcia celulárneho automatu sa získava pomocou genetických algoritmov [24] [7]. Moshe Sipper [29] tieto problémy riešil pomocou neuniformných celulárnych automatov, pre ktoré neplatí, že všetky bunky sa riadia jednou prechodovou funkciou. CA sa dajú použiť aj na také problémy ako je napríklad generovanie prvočísel [35].



Obrázok 2.3: Klasický vs. modifikovaný CA, prevzaté z [5]

Kapitola 3

Evolučný prístup

Medzi najvýznamnejšie poznatky, ktoré ľudstvu dalo 19. storočie, patrí Darwinova evolučná teória. Jej objaviteľ si ako účastník expedície na lodi HMS Beagle všimol rozmanitosti a pritom príbuznosti živičíšnych druhov na rôznych kontinentoch. Predovšetkým pozorovaním ojedinelého ekosystému na Galapágach ho prijmelo, aby si uvedomil, že teologický pohľad na svet so stvorenými formami života je neudržateľný. Vo svojom prevratnom diele „O vzniku druhov prírodným výberom“ definoval pravidlo, podľa ktorého sa život na Zemi vyvinul postupnými zmenami druhov, ktoré nadobudli až dnešných foriem.

Charles Lyell, metodický britský analytik a výzkumník, položil základ pre modernú geológiu. Jeho práce mali veľký vplyv na Darwina, hlavne Lyellovo pojednanie o veku Zeme, na ktorom mohol Darwin stavať svoju teóriu, ktorá vyžadovala vysokú „časovú zložitosť“. Dovtedy panoval názor, že Zem je stará len niekoľko miliónov rokov. Lyell zistil, že naša Zem nie je až taká mladá a odhadol jej vek na niekoľko stoviek miliónov rokov. Ako sa neskôr zistilo, tak aj tento odhad bol milný, dnešné odhady hovoria o pribižne 4 a pol miliarde rokov.

Evolučná teória samozrejme v dobe vydania vyvolala rozruch, hlavne kvôli nekompatibilite s teologickými ideami. Darwinova teória stála aj proti vede, pretože ju nemožno popísať deterministicky ako bolo možné popísať teóriu Lamarckovu, ktorá vysvetľovala schopnosť zmien živých organizmov, a to zdedením štruktúry (napríklad silných svalov) postupne počas mnohých generácií, ktorá sa stáva lepšie vyvinutou v dôsledku trvalého používania alebo naopak horšie vyvinutou vplyvom nečinnosti [28].

Dnes, v dobe po 151 rokoch, technologicky výrazne pokročilejšej, pretrváva množstvo skeptikov tejto teórie. Hlavnou nevýhodou evolučnej teórie je náhoda. Náhoda, ktorá je zodpovedná za mutácie, ktoré zaisťujú vývoj živočíšnych druhov. Evolúcia však v súčasnosti prebieha aj pred našimi očami. Napríklad lieky, ktoré boli úspešné pred niekoľkými desaťročiami, začínajú byť nepoužitelné – toto je výsledkom mutácií, evolúcie. Ale to, že by sa vírusy vyvinuli resp. adaptovali na základe čistej náhody sa zdá mnohým ako nepravdepodobné. Argumenty pre Darwinovu teóriu však ďalekosiahle prevyšujú častokrát nepodložené argumenty skeptikov.

Darwinova evolučná teória bola sformulovaná Darwinom v roku 1859, čo je pomerne neskoro. Je teóriou, ktorá vysvetľuje vývoj druhov, prirodzený výber, na ktorom je evolúcia založená. Súčasné prístupy k tvorbe evolučných návrhov sú viac – menej založené na Darwinovej evolučnej teórií alebo lepšie povedané na tzv. rozšírenej Darwinovej evolučnej teórii, pretože klasická teória Darwina má svoje neresti popísané nižšie. Dnes je teória britského prírodovedca považovaná za predchodcu moderných teórií vysvetľujúcich vznik a vývoj života. Klasická Darwinova teória však nie je dokonalá a časom vyvstali niektoré

problémy:

- prirodzený výber, tak ako ho popísal Darwin, neumožňuje vysvetliť vznik účelných vlastností u pohlavne sa rozmnožujúcich organizmov, Darwin nepoznal mutáciu
- Darwinova teória nevysvetľuje vznik altruizmu (nezištné jednanie v prospech druhých ako mravný princíp)
- dedičnosť vlôh nie je tvrdá, ale mäkká, inými slovami, podľa Darwina sa dedia len vlohy a nie vlastnosti

Princípy evolučnej teórie budeme používať pri návrhu nášho simulátoru, konkrétne princípy genetických algoritmov a evolučných stratégií, ktoré spadajú pod evolučné algoritmy.

3.1 Biologický kontext

V nasledujúcich kapitolách budeme využívať niektoré pojmy prevzaté z biológie, ktoré v krátkosti zhrniem. Nie však vo všetkých prípadoch je biologická sémantika zhodná so sémantikou v rámci evolučných algoritmov.

Medzi základné pojmy Darwinovej teórie patrí populácia. Populácia je množina jedincov. Každý jedinec je reprezentovaný svojím genetickým materiálom. V tejto práci budem voľne zamieňať pojmy genetický materiál, genóm a chromozóm, aj keď z pohľadu biológie tieto pojmy nie sú rovnocenné. Vlastné zakódovanie genetickej informácie do nejakej štruktúry označujeme ako genotyp. Spôsob, akým sa genotyp v danom prostredí interpretuje, sa nazýva fenotyp. Jedinec s rovnakým genotypom môže mať v inom prostredí odlišnú schopnosť prežitia, teda odlišné prostredia spôsobili odlišnú interpretáciu genotypu na fenotyp. Genetický materiál resp. genóm sa skladá z génov. V našom kontexte jeden gén kóduje jednu vlastnosť. Konkrétna hodnota danej vlastnosti, daného génu, sa nazýva alela. V rámci počítačovej terminológie môžeme povedať, že každý gén reprezentuje určitý dátový typ a alely sú hodnotami daného dátového typu, daného génu.

3.2 Zaradenie evolučných algoritmov

Existujú problémy, ktorých presný model sa nedá, nevieme alebo je veľmi náročné zostaviť a navyše, stavový priestor resp. množina riešení daného problému je obrovská. Kvôli takýmto problémom vznikol pojem **SOFTCOMPUTING** a všetko čo podeň spadá. Je to spôsob riešenia problémov „s rozumom“, teda nie hrubou (hard) silou ako je to v prípade klasických algoritmov na prehľadávanie stavového priestoru ako je napríklad BFS (z angl. Breadth – first search) či DFS (z angl. Depth – first search), prípadne ich varianty. Softcomputing rieši problémy pomocou nejakej heuristiky resp. heuristickej funkcie, ktorou algoritmus „myslí“. Potrebné je dodať, že metódy softcomputingu častokrát nedávajú najlepšie riešenie, ale len suboptimálne, ktoré nám však častokrát postačuje. Za softcomputing sa v súčasnosti považuje riešenie komplexného problému pomocou jednou z týchto troch metodológii [28]:

- **Neurónové siete** (NS) – sú inšpirované ľudskými neurónmi. Základy neurónových sietí položili v roku 1943 McCulloch a Pitts [38]. Vďaka tomu, že NS dokážu zachytiť veľa závislostí, dovolujú rýchlo a jednoducho namodelovať skúmaný dej, ktorý by sa inak modeloval veľmi ťažko. Neurónová sieť sa chová ako „čierna skrinka“, čo má svoje výhody aj nevýhody.

- **Evolučné algoritmy (EA)** – sú spoločným vyjadrením pre množinu moderných matematických postupov, ktoré využívajú modely evolučných procesov v prírode založených na Darwinovej evolučnej teórii popísanej na začiatku tejto kapitoly. Jednotlivé riešenia, ktoré tvoria populáciu, sa vyvíjajú na základe klasických evolučných a genetických operátorov ako je selekcia, kríženie či mutácia. EA stavajú a súčasne aj padajú na fitness funkcii. Fitness funkcia definuje schopnosť jedinca prežiť v danom prostredí resp. schopnosť riešenia riešiť daný problém. Evolučné algoritmy sú v poslednej dobe veľmi rozšírené. Prílišná popularizácia so sebou však prináša aj nerealistické očakávania. Podobne ako evolučné algoritmy, tak aj všeobecne všetky optimalizačné techniky nefungujú ako univerzálne metódy, ale každá z nich sa hodí na inú oblasť problémov a zistiť, aká technika je najlepšia, prípadne s akými parametrami, je už úlohou inžinierskeho návrhu.

  Medzi hlavné paradigmá evolučných algoritmov patria:

  - **Genetické Algoritmy (GA)** – za ich priekopníka sa považuje John Holland, ktorý sa zaoberal štúdiom elementárnych procesov v populáciach. Na základe týchto štúdii navrhol genetický algoritmus ako abstrakciu príslušných biologických procesov. Genetické algoritmy pracujú s binárnym genómom, teda každý gén chromozómu môže nadobúdať len dvoch hodnôt, 0 alebo 1. Všetci jedinci majú genóm rovnakej dĺžky.
  - **Genetické Programovanie (GP)** – je možné považovať za rozšírenie genetických algoritmov. Novinkou, ktorú genetické programovanie prináša, je to, že chromozómy už nie sú reťazce pevnej dĺžky, ale hierarchicky štrukturované počítačové programy, ktoré po spustení môžu predstavovať potenciálne riešenie daného problému [15]. Za duchovného otca genetického programovania sa považuje standfordský informatik John Koza, ktorého vynikajúce knihy (napríklad [20]) sa považujú za Bibliu v tejto oblasti. Základným princípom je evolúcia celých programov, ktorých genotypom je akási stromová hierarchická štruktúra, ktorá sa skladá z tzv. terminálov (konštanty, premenné atď.), čo sú listy stromovej štruktúry, a operátorov, ktoré tvoria uzly stromovej štruktúry.

    Na takto definované genómy jedincov sú sú aplikované genetické operátory, ktoré, pretože pracujú s inými štruktúrami, musia byť implementačne odlišné, ale principiálne sú totožné.
  - **Evolučné Programovanie (EP)** – prvýkrát použil Lawrence J. Fogel, genóm je reprezentovaný reálnymi číslami, pri EP sa nevyužíva rekombinácia, ale iba mutácia a selekcia, teda každý rodič vytvorí jedného potomka prostredníctvom mutácie. Mutačný operátor sa implementuje pomocou pripočítania hodnoty podľa Gaussovho rozdelenia pravdepodobnosti [9].
  - **Evolučné Stratégie (ES)** - keď okolo roku 1963 začali Hans-Paul Schwefel a Ingo Rechenberg na Technickej univerzite v Berlíne s napodobnovaním vývoja v prírode, boli presvedčení, že ich metóda nejlepšie napodobňuje evolúciu v živej prírode. Preto svoju metódu nazvali všeobecne evolučné strategie. Postupom času se však ukázalo, že tento spôsob rieši len určitý typ úloh, hlavne v stavebnom a strojnom inžinierstve. Genetické algoritmy nie sú teda podradené evolučným stratégiam, ale naopak ich svojou popularitou zatieňujú [16]. Genóm v rámci evolučných stratégií je zložený z génov, ktoré sú reprezentované reálnymi číslami, z čoho vyplýva implementačná diferenciácia genetických operátorov. Mutačný

operátor sa väčšinou implementuje pomocou pripočítania hodnoty podľa Gaussovho rozdelenia pravdepodobnosti ako pri EP. Takýto spôsob mutácie rieši jeden problém genetických algoritmov – malé zmeny genotypu nemusia viesť k malým zmenám fenotypu[1].

- **Fuzzy logika** – je rozšírením klasickej Booleovskej logiky, rieši jedno veľké obmedzenie klasikej dvojstavovej logiky – dané tvrdenie môžeme ohodnotiť len dvoma hodnotami: pravda – „1“ alebo nepravda – „0“ [27]. Fuzzy logika umožňuje matematicky korektne, pomocou fuzzy množín, zohľadniť neurčitosť, neznalosť, nepresnosť, hlavne vtedy, keď nejde o následky náhodných javov.

  Základom fuzzy logiky sú fuzzy množiny. Fuzzy množina je zovšeobecnením klasickej množiny v tom zmysle, že ide o zobrazenie univerza X na celý interval $< 0, 1 >$. Keďže fuzzy množinu nejde všeobecne popísať inak ako charakteristickou funkciou (funkciou príslušnosti), považujú sa termíny fuzzy množina a charakteristická funkcia za totožné. Formálne zapísanie klasickej a fuzzy množiny:

$$\begin{aligned} \text{Klasická množina}: &\quad f : X \to \{0, 1\} \\ \text{Fuzzy množina}: &\quad f : X \to < 0, 1 > \end{aligned}$$

  Keďže fuzzy logika pracuje s celým intervalom $< 0, 1 >$, môžeme povedať „ako veľmi“ je dané tvrdenie pravdivé. Pomocou tejto logiky môžeme teda definovať aj tzv. vágne alebo lingvistické premenné ako je napríklad „veľa“, „málo“, „akurát“. Napríklad môžeme povedať, že 13 stupňov je 0.7 zima, 0.4 príjemne a 0.1 teplo. Fuzzy logika sa používa hlavne na riadenie a reguláciu.

3.3 Genetické operátory

Variácia genómu je nutná v evolučnom procese. Genetické operátory používané v evolučných algoritmoch sú analogické tým prírodným: prežitie najprispôsobivejšieho resp. selekcia, reprodukcia resp. kríženie alebo rekombinácia a mutácia.

3.3.1 Selekcia

Selekcia je základným evolučným operátorom, bez ktorého použitia by sa nedalo hovoriť o evolučných algoritmoch. Tento operátor reprezentuje prirodzený výber popísaný Darwinom. Rozoznávame 3 najpoužívanejšie typy selekcie a ich varianty:

- **koleso šťastia** – tento typ selekcie funguje tak, že jedincom priradíme pravdepodobnosť výberu do ďalšej generácie podľa hodnoty fitness funkcie. Jedinci s lepšou hodnotou fitness funkcie budú vybraný do ďalšej generácie s vyššou pravdepodobnosťou ako jedinci s horšou hodnotou fitness funkcie.

- **turnaj** – je založený na náhodnom výbere dvojíc a ich súboji na základe hodnôt fitness funkcie. Do ďalšej generácie majú šancu sa dostať aj jedinci, ktorý sú slabší, ale v turnaji stretli ešte slabšieho, slabšieho v rámci hodnôt fitness funkcie.

[1] Tento problém sa dá obísť napríklad pomocou Grayovho kódovania.

- „**najlepší vyhráva**“ – je najjednoduchším typom selekcie, v každej generácii preferuje len tých najstatnejších jedinov, jedincov umiestnených na čelných priečkach rebríčka tvoreného komisiou, ktorá hodnotí statnosť jedinca na základe hodnoty fitness funkcie. Tento typ selekcie je vhodný v prípade, že fitness funkcia nemá veľa extrémov, najlepšie je, keď má len 1 globálny extrém, pretože evolučné algoritmy založené na tomto type selekcie nevedia riešiť tzv. klamné problémy, prípadne multimodálne funkcie, pretože v populácii nezachovávame diverzitu, vyberáme len tých jedincov s najlepšou hodnotou fitness funkcie. Evolučné algoritmy založené na tomto type selekcie častokrát predbežne konvergujú a teda uviaznú v lokálnom extréme.

Pri reálnych praktických prístupoch sa len málokedy stretneme len s čisto jedným druhom selekcie, takmer vždy sa používajú ich možné variácie a kombinácie. Pre priblíženie, existujú aj prístupy, ktoré pracujú s dvoma populáciami kvôli zachovaniu rôznorodosti populácie a dochádza k migrácii medzi populáciami. Každá populácia je však založená na inej fitness funkcii [25].

So selekciou úzko súvisí obnova populácie. Po evaluácii hodnôt fitness funkcie jedincov populácie a selekcii jedincov, ktorí „prežijú“, máme viac možností ako nahradiť aktuálnu populáciu novou. Rozlišujeme 2 základné prístupy:

- **Úplná obnova populácie** – dochádza k vymieraniu rodičov, teda celá generácia je nahradená novou. Nová generácia je vytvorená na zákalde zvoleného spôsobu selekcie jedincov.
- **Čiastočná obnova populácie** (steady state) – potomkami sa nahradí len pár jedincov.

Väčšinou sa používajú varianty, ktoré nahradzujú $20 - 50\,\%$ populácie [28].

3.3.2 Kríženie

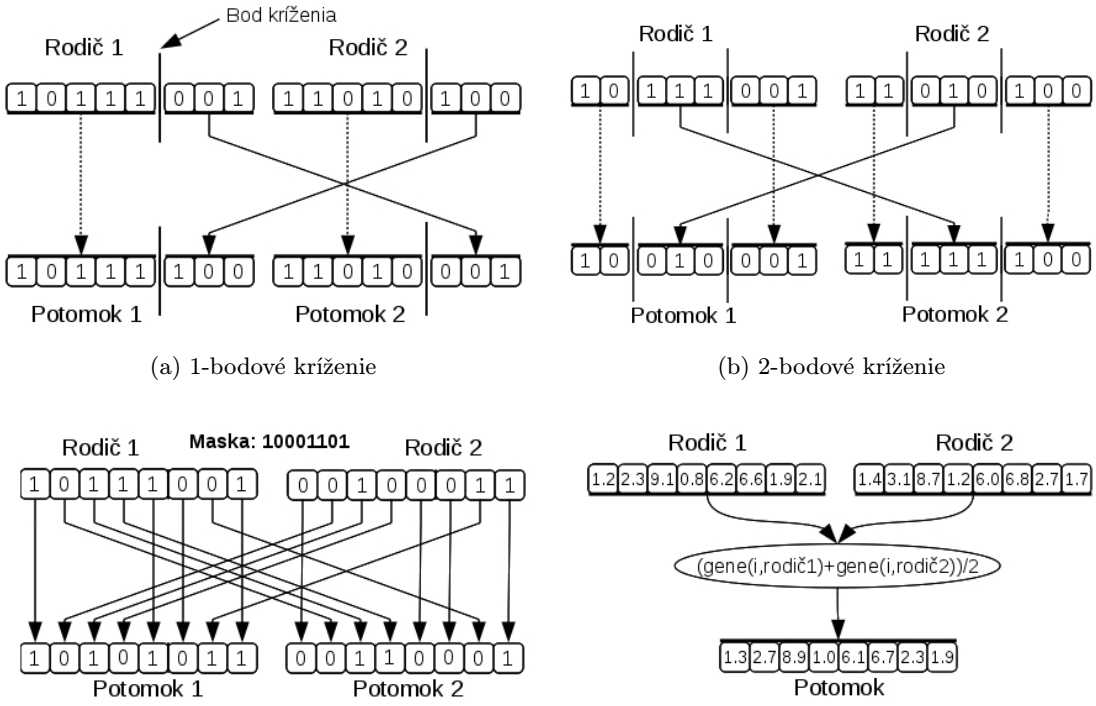
Ku genetickému operátoru kríženia môžeme pristupovať rôznymi spôsobmi:

- **všeobecne n bodové kríženie** – najpoužívanejší typ kríženia, väčšinou sa používa kríženie jednobodové alebo dvojbodové, závisí samozrejme na danom probléme a veľkosti genómu.
- **kríženie maskou** – náhodne sa vygeneruje bitová maska, ktorej dĺžka je zhodná s dĺžkou chromozómu a noví jedinci sa vytvoria tak, že gény na pozíciach obsahujúcich „0“ zdedí prvý potomok a gény na pozíciach obsahujúcich „1“ zdedí potomok druhý.
- **aritmetické kríženie** – sa využíva hlavne pri evolučných stratégiach, kde gény sú reprezentované reálnymi číslami. Noví jedninci sa vytvoria na základe aplikácie nejakého aritmetického operátoru (väčšinou priemer) na gény rodičov.

Na obrázku 3.1 sú graficky znázornené spomínané typy operátoru kríženia.

3.3.3 Mutácia

Mutácia je motorom genetických algoritmov, je založená na náhodnej zmene génu chromozómu jedinca, mutácia zabezpečuje vznik nových vlastností, zmenu génov. Rozlišujeme v zásade 2 typy mutácie:



(a) 1-bodové kríženie

(b) 2-bodové kríženie

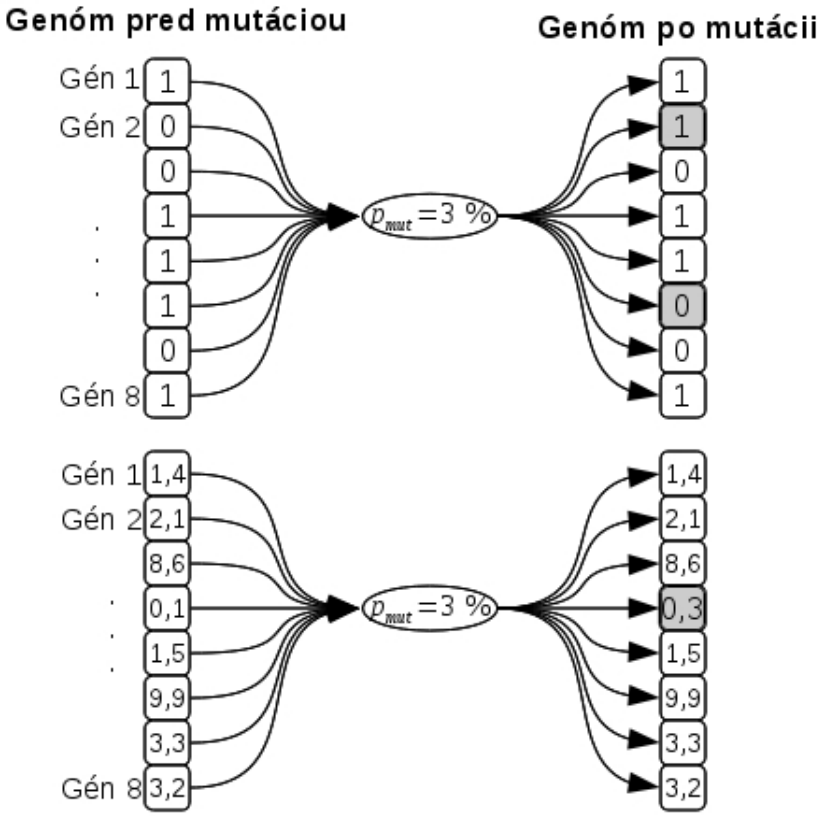
(c) Kríženie maskou

(d) Aritmetické kríženie (pomocou priemeru)

Obrázok 3.1: Rôzne typy evolučného operátoru kríženia

- **inverzia génu** – vhodné a použitelné len pri binárnej reprezentácii génov, teda génoch, ktoré môžu existovať len v dvoch rôznych alelách
- **pripočítanie hodnoty rozloženia pravdepodobnosti** – využíva sa pri reprezentácii génov pomocou reálnych čísel, k hodnote génu sa pripočíta hodnota daná určitým rozdelením pravdepodobnosti, väčšinou sa používa normálne (Gaussovo) alebo uniformné rozdelenie pravdepodobnosti

Obrázok 3.2 ilustruje popísané spôsoby mutácie.



Obrázok 3.2: Mutácia genómu, horná časť obrázku reprezentuje aplikáciu operátoru mutácie pomocou inverzie génu, spodná časť pomocou pripočítania hodnoty rozloženia pravdepodobnosti.

Kapitola 4

Evolučný návrh simulátoru

Ako bolo spomenuté v úvodnej kapitole, cieľom tejto práce je vytvoriť simulátor, ktorý by dokázal predpovedať resp. simulovať chovanie systému v budúcnosti alebo v nasledujúcich krokoch na základe dát z minulosti. Simulátor by mal umožňovať riešiť komplexné problémy alebo javy, ktorých exaktná predpoveď by bola náročná alebo nemožná.

Celý problém predikcie je možné minimalizovať na hľadanie koeficientov zvolenej štatistickej metódy, pomocou ktorej hľadáme vzťahy medzi nezávislými premennými (parametrami buniek CA) a závislou premennou (stav bunky CA). Návrh bude založený na modifikovanom koncepte CA (modifikácia sa týka prechodovej funkcie) v spolupráci s evolučnými stratégiami. Prechodovú funkciu celulárneho automatu bude mať na starosti regresná analýza (súbor štatistických metód), konkrétne lineárna a o niečo sofistikovanejšia logistická regresia. Prečo práve tieto 2 štatistické techniky? Pred odpoveďou si ich však najskôr bližšie predstavíme v nasledujúcej sekcii a argumenty predložíme v sekcii 4.2.

4.1 Regresná analýza

Modelovanie vzťahov medzi vysvetľujúcimi premennými (nezávislými) a vysvetľovanou premennou (závislou) patrí medzi základné aktivity, s ktorými sa v štatistike dá stretnúť. Rozlišujeme 2 typy premenných podľa usporiadania ich možných hodnôt [1]:

- **premenné usporiadaných hodnôt** (ordinal variables) – hodnoty, ktorých môžu tieto premenné nadobúdať, je možné zoradiť (napr. pekne, celkom pekne, normálne, celkom škaredo, škaredo)
- **premenné neusporiadaných hodnôt** (nominal variables) – hodnoty, ktorých môžu tieto premenné nadobúdať, nie je možné zoradiť (napr. premenná reprezentujúca typ prepravy do práce môže mať jednu z hodnôt: autobus, bicykel, lietadlo, vlak). Štatistická analýza nezávisí na poradí hodnôt. Pretože usporiadanie hodnôt je irelevantné, metódy musia dávať rovnaké výsledky pri akomkoľvek zoradení hodnôt.

Metódy stavané pre premenné usporiadaných hodnôt sa nedajú použiť na premenné neusporiadaných hodnôt, ale naopak to neplatí. Z tejto dvojitej negácie vyplýva, že metódy pre tzv. nominálne premenné je možné použiť na premenné hodnôt usporiadaných, ale s potenciálne horším výsledkom ako by sa dosiahlo s nejakou vhodnou metódou, pretože metódy špecializované na tento typ závislej premennej berú do úvahy aj zoradenosť hodnôt.

Existuje zovšeobecnená trieda štatistických modelov, do ktorej sa zahrňuje regresná analýza a tiež tzv. ANOVA (z angl. ANalysis Of VAriance) modely. Hovorí sa o tzv. generalizovaných lineárnych modeloch (GLM – Generalized linear models).

Jednoducho povedané, regresná analýza je označenie štatistických metód, pomocou ktorých hľadáme vzťahy resp. snažíme sa nejakým spôsobom odhadnúť hodnoty závislých premenných na základe premenných nezávislých. Príklad aplikácie regresnej analýzy môžeme vidieť napríklad v lekárstve, kde sa regresná analýza s úspechom využíva na odhadovanie pooperačnej dĺžky života pacientov trpiacich na rakovinu. Na základe skúseností z minulých rokov, kedy sa zhromaždili predoperačné údaje o zdravotnom stave väčšieho počtu pacientov, napríklad veľkosť a typ nádoru, vek pacientov atp. (regresory), ale aj záznamy o dĺžke života po operácii (regresand), je možné pomocou vhodného typu regresnej analýzy (v tomto prípade tzv. Coxovej regresie) vytvoriť vzorec, s pomocou ktorého bude možné pri novom pacientovi na základe znalosti jeho zdravotného stavu odhadnúť strednú hodnotu očakávanej doby prežitia v prípade operácie. Ak je naviac k dispozícii podobná analýza pre pacientov liečených konzervatívne, tak je možné novému pacientovi doporučiť, ktorý spôsob liečby mu v danej situácii dáva nádej na dlhšie prežitie [34].

Nie všetky štatistické modely sú vhodné na všetko. Výber modelu závisí najmä na povahe dát. V prípade, že závislá a aj nezávislé premenné sú spojité, vhodným modelom môže byť klasická lineárna alebo polynomiálna regresia a rovnice tvorené metódou najmenších štvorcov (OLS). Model lineárnej regresie poskytuje účinný nástroj pre analýzu dát a hľadanie vzťahov medzi nimi. Ak tieto vzťahy sú lineárnej povahy, lineárna regresia je ideálny model. Predpoklady tohoto typu regresie však nie sú splnené napríklad v prípade, ak závislá premenná má len 2, 3 alebo jednoducho povedané – nie veľa kategórii. Podľa profesora Scotta Menarda [23], hlavne pri dichotomických (dvojhodnotových) závislých premenných sú porušené predpoklady homeoskedacity, linearity a normality a metóda najmenších štvorcov je prinajlepšom neefektívna. Logistická regresia predchádza týmto problémom zavedením tzv. logitu (prirodzený logaritmus šance patričnosti do určitej kategórie).

4.1.1 Lineárna regresia

Lineárna regresia sa väčšinou používa na popis vzťahov medzi nezávislými premennými a závislou premennou, ktorá je spojitá.

S množinou n štatistických jednotiek $\{y_i, x_{i,1}, x_{i,2}, \ldots, x_{i,p}\}_{i=1}^{n}$, štatistický model lineárnej regresie predpokladá, že vzťah medzi závislou premennou y a nezávislými premennými x_p je lineárny. Model môžeme vyjadriť pomocou vzťahu:

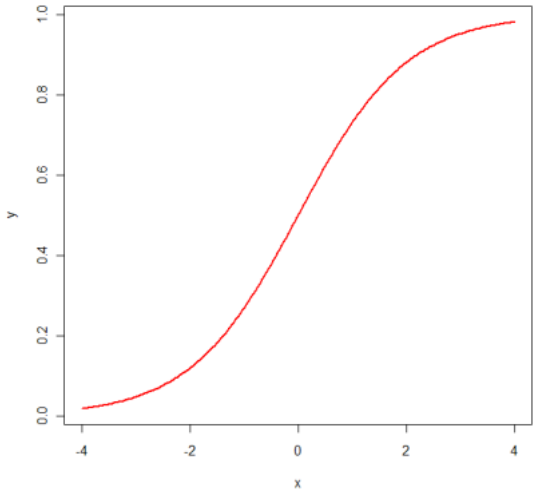
$$y_i = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} = \alpha + \vec{\beta}^{\mathrm{T}} \vec{x} \tag{4.1}$$

kde y_i predstavuje hodnotu závislej premennej na základe hodnôt nezávislých premenných $x_{i1}, x_{i2}, \ldots, x_{ip}$ resp. $\vec{x}$. Váhy $\beta_1, \beta_2, \ldots, \beta_p$ alebo skrátene $\vec{\beta}$ modelujú vlastné závislosti medzi premennými. Konštanta α reprezentuje hodnotu y_i, keď hodnoty všetkých nezávislých premenných sú rovné nule.

4.1.2 Logistická regresia

Logistická regresia sa používa na popis vzťahov medzi určitými triedami v prípade, že závislá premenná nie je spojiým štatistickým znakom, ale znakom binárnym/kategorickým. Alternatívou logistickej regresie je diskriminačná analýza [4]. Logistická regresia je vhodná

v prípade očakávaných nemonotónnych vzťahov medzi nezávislými premennými a premennou závislou. Je pravdepodobnostným modelom, to znamená, že určujeme pravdepodobnosť príslušnosti do určitej kategórie. Z toho vyplýva, že hodnoty logistickej regresie nemôžu presiahnuť interval $< 0, 1 >$. Navyše, tvar krivky výslednej pravdepodobnosti nie je lineárny (viď obrázok 4.1). Tento tvar je špecifický tým, že s veľkou zmenou nezávislých premenných pri hraničných pravdepodobnostiach 0 a 1 sa pravdepodobnosť mení málo. Táto vlastnosť je výhodná pri modelovaní reálnych systémov.



Obrázok 4.1: Logistická krivka pre 1 nezávislú premennú x

Napriek tomu, že logistická regresia patrí pod generalizované lineárne modely (GLM), je schopná modelovať nelineárne vzťahy. Dá sa povedať, že vzťahy sú nelineárne v rámci skúmaných premenných (o čo nám ide), ale formálne sú lineárne v rámci ich parametrov [3]. Existujú 3 hlavné typy logistickej regresie:

- binárna (dichotomická) logistická regresia
- multinominálna logistická regresia
- logistická regresia premenných usporiadaných hodnôt

Binárna (dichotomická) logistická regresia

Binárne dáta sú najčastejšou formou kategorických dát, najpopulárnejším modelom pre binárne kategorické dáta je binárna alebo dichotomická logistická regresia.

Metóda dichotomickej logistickej regresie predpokladá, že za podmienok, ktoré určuje vektor $\vec{x}$ (vektor nezávislých premenných), bude náhodná veličina $Y(\vec{x})$ rovná 1 s pravdepodobnosťou, ktorej závislosť na $\vec{x}$ môžeme vyjadriť pomocou tzv. logistickej funkcie:

$$P[Y(\vec{x}) = 1] = \frac{e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}}{1 + e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}} \tag{4.2}$$

kde $\vec{\beta}$ je vektor váh jednotlivých nezávislých alebo vysvetľujúcich premenných a $\vec{\beta}^{\mathrm{T}}\vec{x}$ je lineárna kombinácia týchto váh a hodnôt nezávislých premenných $\vec{x}$. Výslednú pravdepodobnosť môžeme následne upraviť:

$$P[Y(\vec{x})=1] = \frac{e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}}{1+e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}} = \frac{1}{\frac{1+e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}}{e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}}} = \frac{1}{1+\frac{1}{e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}}} = \frac{1}{1+e^{-(\alpha+\vec{\beta}^{\mathrm{T}}\vec{x})}} \tag{4.3}$$

Ako sa prišlo k pomerne škaredému vzorcu 4.2? V terminológii logistickej regresie rozlišujeme 2 pojmy: pravdepodobnosť a šanca (angl. odds). Existuje vzťah medzi pravdepodobnosťou a šancou javu X:

$$\mathrm{odds}(X) = \frac{P(X)}{1-P(X)} \tag{4.4}$$

Pracujme s alternatívnym rozložením. Pravdepodobnosť, že nastane jav A je $P(A)$, pravdepodobnosť, že tento jav nenastane je $1-P(A)$. Budeme súčasne pracovať s príkladom: pravdepodobnosť, že na klasickej 6–hrannej kocke hodíme 1 je $\frac{1}{6}$, formálne $P(S)=\frac{1}{6}$, kde S je jav hodu šestky. Z pohľadu šancí:

$$\mathrm{odds}(A) = \frac{P}{1-P} \tag{4.5}$$

$$\mathrm{odds}(A') = \frac{1-P}{P} \tag{4.6}$$

$$\mathrm{odds}(S) = \frac{\frac{1}{6}}{1-\frac{1}{6}} = \frac{1}{5} \tag{4.7}$$

$$\mathrm{odds}(S') = \frac{1-\frac{1}{6}}{\frac{1}{6}} = \frac{5}{1} \tag{4.8}$$

Z rovníc 4.7 a 4.8 je možné vidieť, že šanca hodenia kockou šestku je 0.2, šanca nehodenia šestky je 5. Táto asymetria však nie je vhodná, chceme, aby šanca javu A' bola opakom javu A. Toho môžeme dosiahnuť pomocou prirodzeného logaritmu. Prirodzený logaritmus šance závislej premennej sa nazáva sofistikovane **logit**. Teda dostávame:

$$\ln(odds(A)) = \mathrm{logit}(P(A)) = \ln\left(\frac{P(A)}{1-P(A)}\right) \tag{4.9}$$

$$\ln(odds(A')) = \mathrm{logit}(P(A')) = \ln\left(\frac{1-P(A)}{P(A)}\right) \tag{4.10}$$

resp.

$$\ln(odds(S)) = \mathrm{logit}(P(A)) = \ln\left(\frac{\frac{1}{6}}{1-\frac{1}{6}}\right) = \ln\left(\frac{1}{5}\right) \doteq -1.61$$

$$\ln(odds(S')) = \mathrm{logit}(P(A')) = \ln\left(\frac{1-\frac{1}{6}}{\frac{1}{6}}\right) = \ln\left(\frac{5}{1}\right) \doteq 1.61$$

V logistickej regresii pri n nezávislých premenných platí:

$$\mathrm{logit}(P(A)) = \alpha+\beta_1x_1+\beta_2x_2+\beta_3x_3+\cdots+\beta_nx_n = \alpha+\vec{\beta}^{\mathrm{T}}\vec{x} \tag{4.11}$$

Zo vzťahu 4.11 vyplýva, že logit pravdepodobnosti javu A je v lineárnom vzťahu so všetkými nezávislými premennými $x_1, x_2, \ldots, x_n$, teda za logistickou regresiou je skrytá, aj keď len formálne, klasická lineárna regresia. Pracujeme však s logitom, ale my sa potrebujeme prepracovať k pravdepodobnosti: Kedže platí 4.9, získavame:

$$\begin{aligned} \ln\left(\frac{P(A)}{1-P(A)}\right) &= \alpha + \vec{\beta}^{\mathrm{T}}\vec{x} \\ \frac{P(A)}{1-P(A)} &= e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}} \\ P(A) &= \frac{e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}}{1+e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}} = \frac{1}{1+e^{-(\alpha+\vec{\beta}^{\mathrm{T}}\vec{x})}} \end{aligned} \tag{4.12}$$

Môžeme vidieť, že rovnica 4.12 sa rovná rovnici 4.2 na začiatku sekcie. Súčet pravdepodobností oboch kategórii závislej premennej sa musí rovnať 1, takže pre pravdepodobnosť javu A' platí:

$$P(A^{'}) = 1 - P(A) = \frac{1}{1+e^{\alpha+\vec{\beta}^{\mathrm{T}}\vec{x}}} \tag{4.13}$$

Multinominálna logistická regresia

Multinominálna logistická regresia sa snaží modelovať prípady, kedy odhadujeme hodnotu premennej minimálne troch neusporiadaných kategórii (nominal variable) na základe nezávislých premenných. Binárna alebo dichotomická logistická regresia je jej špeciálnym prípadom, pretože pracuje len s 2 kategóriami.

V rámci multinominálnej logistickej regresie ľubovolne určíme jednu kategóriu závislej premennej ako tzv. referenčnú kategóriu k_{ref}. Ostatné kategórie budeme porovnávať s touto referenčnou kategóriou. Výber inej referenčnej kategórie nezmení formu modelu, ale zmení interpretáciu ohodnotenia parametrov modelu [18].

Definujeme logit ako prirodzený logaritmus šance (odds) resp. podielu pravdepodobnosti, že hodnota získaná na základe nezávislých premenných „padne“ do kategórie k_i, a pravdepodobnosti, že hodnota „padne“ do referenčnej kategórie k_{ref}:

$$\mathrm{logit}(k_i) = \ln(odds(k_i)) = \ln\left(\frac{P(k_i)}{P(k_{ref})}\right) = \alpha_i + \vec{\beta_i}^{\mathrm{T}}\vec{x} \tag{4.14}$$

Pravdepodobnosť všetkých kategórii okrem referenčnej môžeme vyjadriť ako:

$$P(k_i) = P(k_{ref}) \cdot e^{\alpha_i + \vec{\beta_i}^{\mathrm{T}}\vec{x}} \tag{4.15}$$

Pri n kategóriach a keď napríklad $ref = n$, súčet pravdepodobností všetkých hodnôt závislej premennej musí byť 1:

$$\sum_{i=1}^{n-1} P(k_i) + P(k_{ref}) = 1 \tag{4.16}$$

Odtiaľ môžeme vyjadriť pravdepodobnosť referenčnej kategórie:

$$P(k_{ref}) = \frac{1}{1 + \sum_{i=1}^{n-1} e^{\alpha_i + \vec{\beta_i}^{\mathrm{T}} \vec{x}}} \tag{4.17}$$

Pravdepodobnosti ostatných hodnôt závislej premennej vyjadríme pomocou pravdepodobnosti referenčnej katórie ako:

$$P(k_i) = \frac{e^{\alpha_i + \vec{\beta_i}^{\mathrm{T}} \vec{x}}}{1 + \sum_{j=1}^{n-1} e^{\alpha_j + \vec{\beta_j}^{\mathrm{T}} \vec{x}}} \tag{4.18}$$

Logistická regresia premenných usporiadaných hodnôt

Tento typ logistickej regresie sa snaží na základe nezávislých premenných odhadovať premennú usporiadaných hodnôt (ordered variable). Scott Menard [23] navrhol niekoľko spôsobov riešenia takéhoto typu logistickej regresie:

1. Zasadiť kategórie na intervalovú škálu; to povedie k metóde minimálnych štvorcov (OLS – ordinary least squares).
2. Použiť váhovanú metódu minimálnych štvorcov (WLS – weighted least squares).
3. Použiť tzv. kumulatívny logit model (CLM – Cumulative logit model).
4. Využiť multinomálnu regresiu, ale za cenu potenciálne horších výsledkov.

4.2 Prechodová funkcia CA

Po priblížení regresnej analýzy v predchádzajúcej sekcii môžeme odpovedať na otázku z úvodu kapitoly, ktorá znela: „Prečo budeme využívať lineárnu a logistickú regresiu?“. Lineárna regresia je najjednoduchším modelom v rámci regresnej analýzy. Výsledná hodnota je lineárnou kombináciou hodnôt nezávislých premenných. Implementačne jednoduchá záležitosť. Má však v rámci nášho evolučného návrhu potenciálne nevýhody: je schopná modelovať len lineárne vzťahy a závislá premenná je spojitá. Linearita však nemusí byť na škodu, hlavne v prípade predikcie systému, v rámci ktorého sa vzťahy javia ako lineárne. Druhú nevýhodu, ktorou je spojitosť závislej premennej, obídeme pomocou tzv. prahu, ktorý si však popíšeme na logistickej regresii. Logistická regresia sa zdá byť rozumnou pre náš návrh, kde nezávislé premenné sú spojité a závislá je kategorická, hlavne v prípade predikcie rastu urbanizácie oblastí [21] [37] a predikcie využitia pôdy [30] sa logistická regresia osvedčila. Nevýhodou logistickej regresie je však o niečo väčšia časová zložitosť ako pri regresii lineárnej, čo môže byť hlavne pri predpokladaných lineárnych vzťahoch kontraproduktívne.

Pri logistickej regresii pracujeme s logitom, ktorý je reprezentovaný lineárnou kombináciou váh nezávislých premenných a ich hodnôt. Logit prináša nelinearitu (hoci nie formálnu), ktorá je výhodná, pretože priestorové komplexné systémy, pre ktoré navrhujeme simulátor, sú väčšinou poprepájané nemonotónnymi vzťahmi. Pomocou nelineárnej krivky na obrázku 4.1 dostávame už spomínanú vlastnosť: čím viac sa hodnoty pravdepodobnosti blížia hraničným hodnotám (0 a 1), tým sa s väčšou zmenou nezávislých hodnôt pravdepodobnosť

mení menej [23]. Pri klasifikácii do kategórii závislej premennej však máme na základe logistickej regresie pravdepodobnosti príslušnosti do určitej kategórie. To však nie je to pravé orechové, preto zavádzame tzv. prah, ktorý nám určí, či pravdepodobnosť prechodu do daného stavu (kategórie závislej premennej) je natoľko veľká, aby do tohto stavu prešla. Prah v prípade logistickej regresie leží v rozsahu $< 0, 1 >$. Určenie hodnoty prahu však nie je jednoduchá záležitosť, môžeme ho určiť na základe skúseností experta alebo sa môže stať jedným z génov chromozómu pri určovaní vhodných váh parametrov buniek CA [37]. Techniku prahovania použijeme aj v prípade návrhu založenom na lineárnej regresii, jediným rozdielom je, že prah nebude z rozsahu $< 0, 1 >$, ale z intervalu daného oborom hodnôt závislej spojitej premennej.

Pri vlastnom návrhu sa budeme inšpirovať prácami, ktoré sa zaoberajú predikciou urbanizácie určitých lokalít [21] [37]. Cieľom návrhu je zovšeobecniť prístupy popísané v spomínaných prácach na ľubovolný priestorový komplexný jav.

Prechodová funkcia alebo súbor pravidiel celulárneho automatu sa výrazne líši od klasického konceptu celulárneho automatu popísaného v kapitole 2.2. Stav bunky v danom časovom okamihu nezávisí len na stave okolitých buniek a aktuálnej bunky, ale aj na rôznych parametroch, ktoré sú definované pre každú bunku bunkového poľa. Počet okolitých buniek v danom stave tvorí len 1 z n možných, potenciálne závislých parametrov. Kedže rôzne parametre môžu nadobúdať rôznych hodnôt z rôznych intervalov, je vhodné hodnoty znormalizovať do intervalu $< 0, 1 >$. Normalizácia hodnôt nám aj zjednoduší následnú interpretáciu váh jednotlivých parametrov. Keďže je veľmi zložité vypočítať alebo odhadnúť hodnoty váh, resp. koeficientov daných parametrov, ktoré by reflektovali dependenciu stavu bunky na danom parametre, budeme tieto koeficienty tzv. kalibrovať, teda optimalizovať resp. hľadať najlepšie možné koeficienty, ktoré by čo možno najlepšie modelovali vzťahy v skúmanom systéme. Z týchto váhovaných parametrov sa vypočíta hodnota v rozsahu $< 0, 1 >$ v prípade logistickej regresie, alebo v inom rozsahu popísanom nižšie v prípade regresie lineárnej. Prah následne rozhodne o tom, či bunka prejde do daného stavu alebo nie. Každý stav (kategória) má svoj vlastný prah kvôli lepšiemu popisu vzťahov. Pri predikcii nasledujúceho stavu bunky sa berie do úvahy kategória resp. stav s najvyššou pravdepodobnosťou a táto najvyššia pravdepodobnosť sa následne porovná s prahom.

Na kalibráciu váh jednotlivých parametrov a prahov využijeme schopností evolučných algoritmov. Formálny zápis prechodovej funkcie CA návrhu využívajúceho linárnu regresiu môže vyzerať nasledovne:

$$S_{Linear}(c_{i,t+1}) = \begin{cases} s_{new}, & \text{ak } \max_k(\alpha_k + \vec{\beta}_k^{\mathrm{T}}\vec{x}) > p_k \\ s_{old}, & \text{ak } \max_k(\alpha_k + \vec{\beta}_k^{\mathrm{T}}\vec{x}) <= p_k \end{cases} \tag{4.19}$$

kde $S(c_{i,t+1})$ je stav i-tej bunky $c_{i,t+1}$ v čase $t+1$, p_k je prah prechodu bunky do stavu k. Podobne definujeme prechodovú funkciu založenú na logistickej regresii:

$$S_{Logistic}(c_{i,t+1}) = \begin{cases} s_{new}, & \text{ak } \max_k \frac{e^{\alpha_k + \vec{\beta}_k^{\mathrm{T}}\vec{x}}}{1 + \sum_{j=1}^{n-1} e^{\alpha_j + \vec{\beta}_j^{\mathrm{T}}\vec{x}}} > p_k \\ s_{old}, & \text{ak } \max_k \frac{e^{\alpha_k + \vec{\beta}_k^{\mathrm{T}}\vec{x}}}{1 + \sum_{j=1}^{n-1} e^{\alpha_j + \vec{\beta}_j^{\mathrm{T}}\vec{x}}} <= p_k \end{cases} \tag{4.20}$$

kde n je počet stavov. Referenčná kategória, ktorá je potrebná v modeli logistickej regresie, môže byť ľubovolná. Pre jednoduchosť berieme ako referenčnú kategóriu vždy tú

poslednú (definovanú ako poslednú v konfiguračnom súbore), z toho dôvodu sa v prechodovej funkcii 4.20 v menovateli v sume vyskytuje ako najvyšší index výraz $n-1$, pretože n-tá kategória je referenčná a s tou sa pri výpočte pravdepodobností nepracuje.

Uvediem príklad výpočtu stavu jednej bunky celulárneho automatu: majme CA, ktorého bunky môžu byť v 1 z 3 stavov. Každá bunka má vopred definované 3 parametre:

Parameter č.	1	2	3
Hodnota	$0,6$	$0,9$	$0,2$

a pomocou evolučných stratégii sme dostali takéto koeficienty:

Stav i	α_i	$\beta_{i,1}$	$\beta_{i,2}$	$\beta_{i,3}$	γ_{Log}	γ_{Lin}
1	$0,5$	$0,3$	$-0,1$	$0,8$	$0,44$	$0,6$
2	$0,3$	$0,4$	$-0,2$	$0,6$	$0,7$	$2,4$
3	$0,4$	$0,6$	$0,5$	$-0,4$	$0,9$	$3,1$

Výpočet pomocou lineárnej regresie:

$$\begin{aligned} v_1 &= \alpha_1 + \vec{\beta_1}^{\mathrm{T}}\vec{x} = 0,5 + 0,3 \cdot 0,6 - 0,1 \cdot 0,9 + 0,8 \cdot 0,2 = 0,75 \\ v_2 &= \alpha_2 + \vec{\beta_2}^{\mathrm{T}}\vec{x} = 0,3 + 0,4 \cdot 0,6 - 0,2 \cdot 0,9 + 0,6 \cdot 0,2 = 0,48 \\ v_3 &= \alpha_3 + \vec{\beta_3}^{\mathrm{T}}\vec{x} = 0,4 + 0,6 \cdot 0,6 + 0,5 \cdot 0,9 - 0,4 \cdot 0,2 = 1,13 \end{aligned}$$

Stav s najvyššou hodnotou je stav 3, porovnáme s prahom: $v_3(1,13) < \gamma(3,1)$, teda bunka neprejde do stavu 3, ale zostane v pôvodnom stave.
Výpočet pomocou logistickej regresie (ako referenčný stav resp. kategóriu si zvolíme stav 3):

$$\begin{aligned} w_1 &= \frac{e^{\alpha_1+\vec{\beta_1}^{\mathrm{T}}\vec{x}}}{1 + e^{\alpha_1+\vec{\beta_1}^{\mathrm{T}}\vec{x}} + e^{\alpha_2+\vec{\beta_2}^{\mathrm{T}}\vec{x}}} = \frac{e^{0.75}}{1 + e^{0.75} + e^{0.48}} \doteq 0.447 \\ w_2 &= \frac{e^{\alpha_2+\vec{\beta_2}^{\mathrm{T}}\vec{x}}}{1 + e^{\alpha_1+\vec{\beta_1}^{\mathrm{T}}\vec{x}} + e^{\alpha_2+\vec{\beta_2}^{\mathrm{T}}\vec{x}}} = \frac{e^{0.48}}{1 + e^{0.75} + e^{0.48}} \doteq 0.341 \\ w_3 &= \frac{1}{1 + e^{\alpha_1+\vec{\beta_1}^{\mathrm{T}}\vec{x}} + e^{\alpha_2+\vec{\beta_2}^{\mathrm{T}}\vec{x}}} = \frac{1}{e^{0.75} + e^{0.48}} = 1 - w_1 - w_2 \doteq 0.212 \end{aligned}$$

Stav s najvyššou hodnotou je stav 1, porovnáme s prahom: $w_1(0.447) > \gamma(0.44)$, teda bunka prejde do stavu 1.

4.3 Verzia evolučných algoritmov

Vlastné kalibrovanie resp. hľadanie čo možno najideálnejších váh jednotlivých parametrov buniek budú zabezpečovať evolučné stratégie (ES), ktoré spadajú pod evolučné algoritmy (EA). Dá sa povedať, že ES sú takou odnožou alebo rozšírením genetických algoritmov (GA). EA sa líšia od GA len v tom, že gény chromozómu sú reprezentované reálnymi číslami a s tým súvisí odlišná implementácia niektorých evolučných a genetických operátorov (napr. mutácie). Definujeme chromozóm jedincov populácie, ktorý bude reprezentovať riešenie problému kalibrácie váh pri návrhu s lineárnou regresiou:

$$C_{Linear} = [\alpha_i, \beta_{1,i}, \beta_{2,i}, \ldots, \beta_{k,i}, \gamma_i] \tag{4.21}$$

kde k je počet nezávislých premenných popisujúcich daný jav, $i = 1, 2, \ldots, n$, kde n je počet možných stavov bunky CA. Podobne bude vyzerať chromozóm pre kalibráciu váh logistickej regresie s tým rozdielom, že chromozóm bude o sadu váh jedného stavu (referenčného) kratší, pretože pravdepodobnosť referenčného stavu sme schopný dopočítať. Prah tohto referenčného stavu sa však vyvíjať bude.

$$C_{Logistic} = [\alpha_j, \beta_{1,j}, \beta_{2,j}, \ldots, \beta_{k,j}, \gamma_j, \gamma_n] \tag{4.22}$$

kde index j je definovaný ako $j = 1, 2, \ldots, n-1$, kde n je počet možných stavov bunky CA. V chromozóme sa teda nachádzajú koeficienty $n-1$ rovníc logistickej regresie a n prahov (γ), prah aj pre posledný referenčný stav (kategóriu závislej premennej).

Ako som spomínal, gény chromozómu budú reprezentované hodnotami reálneho dátového typy. Jeden gén môže počas kalibrácie nadobudnúť hodnotu len z intervalu $< 0, 1 >$. Prahy γ_j a γ_n budú môcť nadobudnúť hodnoty z intervalu $< 0, 1 >$, hodnoty prahov γ_i môžu byť z intervalu $< -r, r >$, kde $r = \max \alpha_i + \max \vec{\beta}_i^{\mathrm{T}} \vec{x} = 1 + k$, kde k je počet parametrov buniek CA. Čo sa týka parametrov buniek CA, tak parametre počtu okolitých buniek v určitom stave sú implicitné, teda nie je potrebné ich nejakým spôsobom špecifikovať. Každý stav má toľko parametrov určujúcich počet buniek v určitom stave v definovanom okolí, koľko je celkový počet stavov. Každý z týchto parametrov určuje počet buniek v inom stave. Výnimku tvorí CA, ktorého bunky môžu nadobúdať jednu len z dvoch stavov, kedy sa implicitne vytvorí len jeden (a nie dva) parameter pre popis počtu okolitých buniek v určitom stave. Je celkom zbytočné modelovať napríklad počet zalesnených buniek v okolí aktuálnej bunky a súčasne aj počet buniek nezalesnených, pretože modelujú v podstate to isté, len inak. Pre konkrétny problém bude možné zvoliť vhodné typy genetických operátorov. Prehľad spôsobov implementácie genetických operátorov môžete vidieť v tabuľke 4.1.

Genetický operátor	**Spôsoby implementácie**
Selekcia	ruleta, turnaj, „najlepší vyhráva“
Kríženie	1,2,3-bodové, maskou, aritmetické
Mutácia	Gaussovo, uniformné rozdelenie
Náhrada populácie	čiastočná (steady state), úplná

Tabuľka 4.1: Rôzne spôsoby implementácie genetických operátorov

4.3.1 Fitness funkcia

Menšie hodnoty fitness funkcie v našom návrhu interpretujeme ako lepšie, teda snažíme sa hodnoty fitness funkcie minimalizovať. Fitness funkcia chromozómu C_{Linear} (4.21) a $C_{Logistic}$ (4.22) je zhodná a má nasledujúci tvar:

$$f(C_{Linear}) = f(C_{Logistic}) = \sum_{j=1}^{m-1} \sum_{i=1}^{n} g(S(c_{ij}^{reg}), S(c_{ij}^{data})) \tag{4.23}$$

kde $S(c_{ij}^{reg})$ je stav bunky vypočítaný na základe lineárnej alebo logistickej regresie, $S(c_{ij}^{data})$ je stav bunky podľa získaných dát, n je počet buniek CA, m je počet dostupných trénovacích konfigurácii CA a

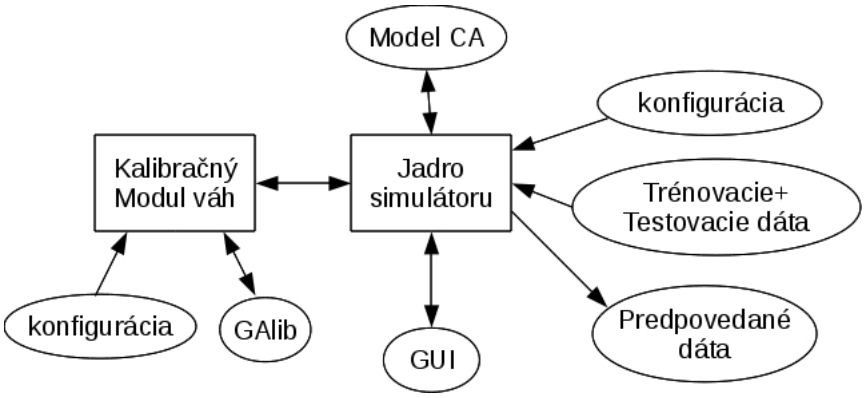
$$g(S(c_{ij}^{reg}), S(c_{ij}^{data})) = \begin{cases} 0, & ak\ S(c_{ij}^{reg}) = S(c_{ij}^{data}) \\ 1, & ak\ S(c_{ij}^{reg}) \neq S(c_{ij}^{data}) \end{cases} \tag{4.24}$$

Ak bude rozdiel časov 2 nasledujúcich trénovacích stavov CA väčší než 1, je nutné vytvárať tzv. medzistavy, ktoré sa do fitness funkcie započítavať nebudú, započítavať sa bude až stav, ktorého čas sa zhoduje s časom nasledujúceho trénovacieho stavu CA. Tvorba medzistavov slúži na zachovanie pravidelného časového intervalu medzi jednotlivými stavmi CA.

Kapitola 5

Implementácia simulátora

Implementácia simulátoru predstavuje prepis návrhu systému do podoby núl a jedničiek, do podoby inštrukcií procesora. Tento prepis je vcelku priamočiary proces. Ide najmä o hľadanie čo najvhodnejších implementačných prostriedkov, ktoré budú čo najlepšie reprezentovať navrhnutý model simulátoru. Schematický popis simulátoru na obrázku 5.1 rozložením modelu na menšie časti je príkladom programovacej techniky zvanej „rozdeluj a panuj“ (divide and conquer).



Obrázok 5.1: Schematické znázornenie simulátoru

5.1 Implementačné prostriedky

Simulátor je implementovaný v programovacom jazyku C++. Keďže existuje množstvo knižníc na prácu s evolučnými algoritmami, na tvorbu evolučného modelu, ktorý bude slúžiť na kalibráciu váh jednotlivých parametrov buniek CA, som použil rozhranie knižnice GAlib [32]. GUI (z angl. Graphical User Interface) je implementované pomocou frameworku Qt, verzia 4.6.3.

5.2 Konfigurácia simulátoru

Rozhraním simulátoru pre prácu s dátami je formát XML. Vstupom simulátoru je okrem konfigurácie aj popis vlastností (parametrov) a stavov buniek CA a vlastné trénovacie

a testovacie dáta. DTD (Document Type Definition) alebo definícia dokumentu (v našom prípade XML dokumentu) popisu parametrov a vlastných dát sa nachádza v prílohe A.

Konfigurácia simulátoru je reprezentovaná klasickým konfiguračným súborom (viď obrázok 5.2), parametrami príkazového riadku pri verzii bez grafického užívateľského prostredia (viď obrázok 5.3) alebo vo verzii s ním (viď obrázok 5.4). Je možné kombinovať konfiguráciu pomocou konfiguračného súboru s konfiguráciou pomocou parametrov príkazového riadka s tým, že vlastnosti nakonfigurované pomocou príkazového riadku majú väčšiu prioritu.

```
## GENERAL CONFIGURATION
# dimenzie celulárneho automatu
dimension = 2

# počet stavov, ktoré môžu bunky nadobúdať
states = 4

# veľkosť CA
dim = 150 # dim1
dim = 200 # dim2

# veľkosť okolia, neighborhood^dimension
neighborhood = 9
```

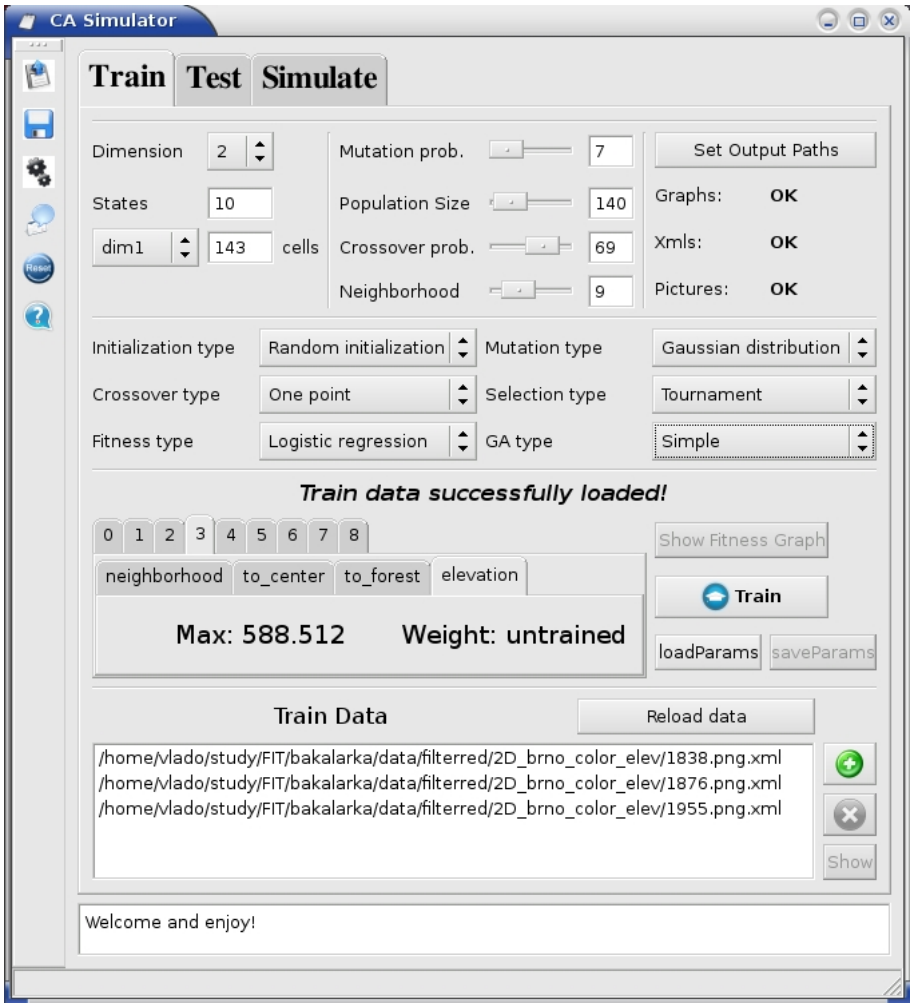
Obrázok 5.2: Ukážka časti konfiguračného súboru

```
./cagasim --pop 100 --pmut 10 --pcross 70 --neigh 11 --stats dir1 --output dir2
          --graphs dir3 --pngs dir4 --xmls dir5 --maxgen 1000 --type-ga 52
          --type-fitness 2 --type-init 11 --type-selection 41
          --type-crossover 21 --type-mutation 31
```

Obrázok 5.3: Ukážka konfigurácie simulátoru pomocou príkazového riadku

5.3 Schopnosti simulátoru

Simulátor je dostupný v 2 verziách – s užívateľským rozhraním a bez neho. Celé užívateľské prostredie sa skladá z menu po ľavej strane, z troch tabov (trénovanie, testovanie, simulácia) a oblasti na výpis stavových informácii (viď obrázok 5.4). V rámci simulátoru je možné nakonfigurovať 1D, 2D alebo 3D model celulárneho automatu, vyššie dimenzie simulátoru nepodporuje. 1-dimenzionálne a 2-dimenzionálne dáta je možné zobraziť a uložiť ako obrázok v png [26] formáte . Výstupy 3D modelov je možné uložiť len ako XML súbor podľa DTD v prílohe. Prechodová funkcia sa zisťuje na základe linárnej alebo logistickej regresie. Simulátor načítava všetky trénovacie dáta pri trénovaní a testovacie dáta pri testovaní do pamäte, čo má výhodu z pohľadu rýchlosti, ale pri väčšom počte buniek CA zaberá veľké množstvo pamäte. Simulátor nie je optimalizovaný na paralelný výpočet. Podrobnejší popis simulátoru sa nachádza v dokumentácii k programu na dvd k práci, prípadne po zadaní prepínača `--help` pri verzii bez GUI alebo v menu GUI verzie.



Obrázok 5.4: Ukážka konfigurácie simulátoru pomocou GUI

Kapitola 6

Predikcia rastu urbanizácie Brna

V dobe, kedy je voľnej pôdy čoraz menej a menej, má význam sa zaoberať rozrastaním miest resp. oblastí. V súčasnosti nastáva rozmach geografických informačných systémov (GIS) hlavne vďaka rýchlemu rozvoju informačných technológií. GIS je počítačový informačný systém pre získavanie, ukladanie, analýzu a vizualizáciu dát, ktoré majú priestorový vzťah k povrchu Zeme [33].

Jednou z vlastností GIS je neschopnosť zachytiť dynamiku dát, na základe ktorej by bolo možné predpovedať stav mesta alebo oblasti o niekoľko rokov alebo desaťročí. Treba dodať, že hľadanie akejsi dynamiky alebo vzťahov v dátach je veľmi náročný proces, pretože rast urbanizácie je komplexný jav, ktorý závisí na obrovskom počte rôznych vplyvov, či už geologických, politických alebo sociologických. Keby sme chceli seriózne simulovať nejaký komplexný priestorový systém ako je napríklad predikcia urbanizácie mesta, dospeli by sme k záveru, že celkový počet parametrov vplývajúcich na tento problém sa môže pohybovať rádovo v stovkách. Navyše, väčšina z nich by bola navzájom previazaná nelineárnymi vzťahmi.

Predikcia urbanizácie na základe konceptu celulárneho automatu je vo veľkej miere študovaným problémom. Najväčšie starosti robí získanie váh jednotlivých parametrov buniek CA tak, aby sa výsledky zhodovali s historickými dátami, čo môže trvať veľmi dlho, pretože prehľadávací stavový priestor je väčšinou obrovský. Tieto váhy sú vložené do nejakého štatistického alebo iného modelu, pomocou ktorého predikujeme stav bunky CA. Kvôli získaniu čo najlepších váh sa začali používať optimalizačné techniky, ktoré sú schopné nejakým spôsobom nájsť suboptimálne riešenie s rozumnou časovou zložitosťou.

Metódy založené na kalibrácii váh parametrov buniek CA môžu využívať rôzne kalibračné techniky. Sú to napríklad neurónové siete [12] [22] [36], evolučné algoritmy [37] [21], či využitie fuzzy logiky [2].

Alternatívou pre techniky založené na kalibrácii váh je technika využívajúca bayesovské siete (Bayesian Networks) [19], ktoré sú schopné pomerne dobre definovať priestorové vzťahy na základe pravdepodobnostného modelu, kde prechodová funkcia je generovaná z grafického formalizmu. Parametre resp. vlastnosti buniek CA sú reprezentované uzlami a vzťahy medzi nimi sú vyjadrené podmienenými pravdepodobnosťami získanými z historických (trénovacích) dát. Nehľadáme najvhodnejšie váhy parametrov, ale najvhodnejšie podmienené pravdepodobnosti medzi jednotlivými uzlami siete.

Prechodovú funkciu budeme tvoriť na základe aj lineárnej, aj logistickej regresie. Pri riešení nášho problému použijeme oba navrhnuté typy prechodovej funkcie a zhodnotíme plusy a mínusy. Experimenty so simulátorom teda budú pozostávať z dvoch častí. V prvej aplikujeme prechodovú funkciu CA založenú na logistickej regresii, v druhej použijeme regresiu

lineárnu. Problémom častokrát je správne určiť rôzne parametre simulátoru. Variabilný bude v našich experimentoch časový rozdiel medzi 2 nasledujúcimi konfiguráciami CA, meniť budeme aj veľkosť okolia, v ktorom budeme počítať zastavané bunky.

6.1 Spracovanie dát

Zdrojom máp pre predikciu bola práca, ktorá študuje zmeny využitia pôdy Brna [8]. Táto štúdia obsahuje mapy Brna a okolia z rokov 1838, 1876, 1955, 1990 a 2005, ktoré použijeme na kalibráciu váh a na testovanie vzniknutej prechodovej funkcie.

Mapy bolo nutné previesť do XML súborov, s ktorými pracuje simulátor. Originálne mapy obsahujú 10 rôznych spôsobov využitia pôdy. Filtrom boli tieto mapy prevedené do „2-stavovej“ čiernobielej podoby, kde každý pixel značí zastavanosť (čierna) alebo nezastavanosť (biela) daného pixelu. Mapy mali rozlíšenie 429x434 pixelov, kvôli zníženiu časovej zložitosti trénovania resp. kalibrácie váh parametrov buniek CA a kvôli nedostatku dát parametra nadmorskej výšky boli mapy navzorkované do konečného rozlíšenia 143x144 pixelov. V prípade dostatku dát by bolo možné pôvodné rozlíšenie ponechať a kvôli zníženiu časovej zložitosti hľadať vhodné váhy na základe navzorkovaných pixelov.

Podarilo sa mi získať dáta 4 vlastností buniek, čo je celkom málo, ale na našu minipredikciu to stačí. Prehľad môžete vidieť v tabuľke 6.1. Počet určitých buniek v definovanom okolí, vzdialenosť od centra Brna a najkratšia vzdialenosť od lesov boli vypočítané z dostupných máp. Nadmorská výška bola získaná pomocou Google Earth Elevation API.

č.	**Parameter**
1	Počet určitých buniek v definovanom okolí
2	Vzdialenosť od centra Brna
3	Najkratšia vzdialenosť od lesov
4	Nadmorská výška

Tabuľka 6.1: Parametre buniek CA

6.2 Experimenty

Nastavenia parametrov simulátoru sú uvedené v tabuľkách 6.2 a 6.3. Rozptyl pri Gaussovom rozložení pravdepodobnosti, ktorého hodnoty využíva mutačný operátor, je 0,2. Budeme pracovať s okolím 5, 7, 9, 11, 13 a 15. Budú skúmané aj časové rozdiely medzi 2 nasledujúcimi konfiguráciami CA, konkrétne časový rozdiel 35 rokov, 15 rokov, 5 rokov a 1 rok. Pri všetkých časových rozdieloch okrem 35 rokov búdú použité na hľadanie optimálnych váh parametrov mapy z rokov 1838, 1876, 1955 a 1990. Mapa z roku 2005 slúži ako testovacia mapa resp. konfigurácia CA. V prípade časového rozdielu 35 rokov, časový odstup máp z rokov 1990 a 2005 však nie je dostatočný, preto bola mapa z roku 2005 z príslušných experimentov vylúčená a mapa z roku 1990 nebude slúžuť na trénovanie, ale testovanie. V tabuľke 6.4 môžete vidieť počet aplikácii prechodovej funkcie na počiatočnú konfiguráciu (rok 1838) potrebných na prechod do stavu v určitom roku, v závislosti na časovom rozdieli medzi 2 nasledujúcimi konfiguráciami CA. Zatiaľ čo v prípade 35-ročného rozdielu aplikujeme prechodovú funkciu 4-krát a sme v roku 1990, v prípade 1 – ročného rozdielu musíme

prechodovú funkciu CA aplikovať až 152-krát. Súčasne budeme merať čas vykonávania evolučného algoritmu, konkrétne priemerný čas evaluácie a selekcie v rámci jednej generácie populácie. Experimenty budú vykonávané na notebooku s Intel Core 2 Duo CPU 2,26 GHz a s operačným systémom Debian s jadrom Linux 2.6.32.

Nastavenie celulárneho automatu			
Dimenzia	Počet stavov	Počet buniek	Počet parametrov
2	2	143x144 = 20 592	4

Tabuľka 6.2: Nastavenie parametrov celulárneho automatu

Nastavenie evolučného algoritmu	
Veľkosť populácie	150
Pravdepodobnosť kríženia	80 %
Pravdepodobnosť mutácie	5 %
Typ selekcie	turnaj
Obnova populácie	úplná

Tabuľka 6.3: Nastavenie parametrov evolučného algoritmu

Časový rozdiel	**1838**	**1876**	**1955**	**1990**	**2005**
35 rokov	0	1	3	4	–
15 rokov	0	2	7	9	10
5 rokov	0	7	23	30	33
1 rok	0	38	117	152	167

Tabuľka 6.4: Počet aplikácii prechodovej funkcie na počiatočnú konfiguráciu, aby sme sa dostali do stavu v určitom roku, v závislosti na časovom rozdiele medzi 2 nasledujúcimi konfiguráciami CA

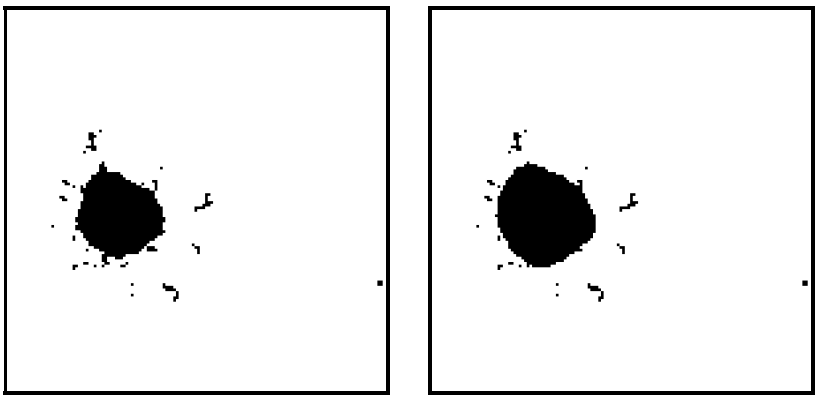
6.2.1 Použitie logistickej regresie

V tabuľke 6.5 sú zobrazené výsledky všetkých experimentov, ktoré modelovali prechodovú funkciu pomocou logistickej regresie. Pri jednotlivých časových odstupoch, ktoré reprezentuje jedna aplikácia prechodovej funkcia CA, sú zaznamenané percentuálne úspešnoti porovnania testovacej a predikovanej testovacej konfigurácie. Znázornená je úspešnosť správnej predikcie buniek, ktoré mali byť v testovacej konfigurácii zastavané a aj úspešnosť predikcie buniek, ktoré zastavané byť nemali. Vypočítaná je aj celková úspešnosť natrénovanej prechodovej funkcie. Pri jednotlivých experimentoch je uvedený priemerný čas, ktorý bol potrebný na evaluáciu, selekciu a obnovu v rámci jednej generácie populácie evolučného algoritmu. Pomlčky značia, že daný experiment nebol vykonaný z dôvodu vysokej časovej náročnosti experimentu. V experimente s časovým rozdielom 35 rokov sme prechodovú funkciu tvorili len na základe 3 trénovacích máp, konkrétne z rokov 1838, 1876 a 1955.

Mapu z roku 2005 sme nemohli využiť ako testovaciu, pretože rozdiel medzi mapou z roku 1990 a z roku 2005 je len 15 rokov. Preto sú aj hodnoty fitness funkcie pri tomto časovom rozdiele nižšie. Z výsledkov vidno, že lepšie hodnoty celkovej úspešnosti majú skôr menšie okolia, modelovanie väčších okolí sa zdá byť kontraproduktívne (viď obrázok 6.1). Čo sa týka rôzneho časového rozdielu 2 nasledujúcich konfigurácii CA, lepšie výsledky vykazujú časové rozdiely 15 a 5 rokov ako rozdiely 35 rokov resp. 1 rok. To je zrejme spôsobené zlou schopnosťou CA modelovať veľké (35 rokov) alebo príliš malé (1 rok) zmeny urbanizácie.

6.2.2 Použitie lineárnej regresie

Výsledky experimentov využívajúcich lineárnu regresiu sú znázornené v tabuľke 6.6. Aj v tomto prípade sa lepšie javia skôr menšie okolia a časové rozdiely medzi jednotlivými konfiguráciami 15 a 5 rokov.



(a) simulovaný stav v r. 1990 (b) simulovaný stav v r. 2025

Obrázok 6.1: Kontraproduktivita veľkého okolia CA

6.3 Zhodnotenie výsledkov

Na základe experimentov môžeme určiť víťazov, jedného z týmu prechodových funkcii založených na logistickej regresii a jedného z týmu prechodových funkcii založených na regresii lineárnej. Najlepšia prechodová funkcia z pohľadu logistickej regresie je založená na 5-okolí a časovom rozdiele 5 rokov. Natrénované koeficienty sú zobrazené v tabuľke 6.7. Môžeme vidieť, že urbanizácia podľa týchto váh závisí maximálne na počte zastavaných buniek v okolí, nepriamo úmerne vplýva na zastavanosť vzdialenosť od centra Brna, teda čím ďalej od centra sa daná bunka CA resp. pixel nachádza, tým je menšia šanca, že sa daná bunka zastavá. Z ďalších dvoch váh vyplýva, že vzdialenosť od lesov a nadmorská výška majú len veľmi malý vplyv na urbanizáciu v lokalite Brna.

Najlepšia prechodová funkcia z pohľadu lineárnej regresie je založená na 9-okolí a časovom rozdiele 15 rokov. V tabuľke 6.8 sú zobrazené kalibrované koeficienty. Opäť, stav zastavania bunky ($\beta_{Urban,1}$) maximálne závisí na počte zastavaných buniek v okolí, ale v prípade zmeny zastavanej bunky na nezastavanú závisí počet zastavaných buniek len polovične ($\beta_{Nonurban,1}$). Zaujímavé sú však hodnoty váh nadmorskej výšky ($\beta_{Urban,4}$ a $\beta_{Nonurban,4}$). Závislosť zastavania bunky od tohto parametra by sme asi odhadovali inú. Vysvetlením môže byť fakt, že Brno a jeho blízke okolie nie je hornatou oblasťou s veľkými zmenami

nadmorskej výšky, takže na zastavanie nevplýva negatívne. Niektoré ďalšie koeficienty sa však dajú interpretovať horšie, napríklad kladná závislosť zastavania bunky na vzdialenosti od centra. Tento jav môže byť spôsobený pádom do lokálneho minima v rámci evolučného algoritmu.

Graf 6.3 poskytuje porovnanie úspešnosti najlepších prechodových funkcii založených na logistickej regresii, lineárnej regresii a prechodovej funkcie, ktorá nemení konfiguráciu CA. Najlepšie prechodové funkcie sú vybrané z množiny všetkých natrénovaných v rámci určitého časového rozdielu konfigurácii CA, teda veľkosť okolia nemusí byť pre všetky prechodové funkcie rovnaká. Môžeme vidieť paradoxnú situáciu, že pasívna prechodová funkcia, ktorá počiatočnej konfigurácii nezmení ani vlások na hlave, je (okrem časového rozdielu 35 rokov) lepšia v rámci porovnania simulovanej a originálnej testovacej konfigurácie. Tento jav je spôsobený poklesom urbanizácie v rozpätí rokov 1990 a 2005. Keďže sme nemodelovali žiadne sociopolitické, prípadne iné zmeny, simulátor tento pokles mohol len veľmi ťažko zachytiť. Napríklad zmena režimu v roku 1989 má na urbanizáciu veľký vplyv. Nemodelovali sme ani žiadne obmedzujúce opatrenia, napríklad obmedzenia závislé na veľkosti populácie. Čo sa týka porovnania 2 typov regresie, logistickej a lineárnej, úspešnosti sú približne rovnaké, čo je dané hlavne lineárnou povahou parametrov buniek CA, teda logistická regresia sa nemohla ukázať v plnej sile a v našej predikcii bola viacmenej kontraproduktívna, pretože je časovo náročnejšia.

Konvergencia evolučného algoritmu bola vo všetkých experimentoch podobná, graf 6.3 znázorňuje konvergenciu EA najlepšieho riešenia, teda prechodovej funkcie založenej na lineárnej regresii s okolím 11 a časovým rozdielom konfigurácii 15 rokov.

Na obrázku 6.4 sa nachádza testovacia konfigurácia, simulovaná testovacia konfigurácia pomocou 2 najlepších prechodových funkcii a simulácia urbanizácie Brna na rok 2020.

V tabuľkách znázorňujúcich experimenty 6.5 a 6.6 sa okrem iného nachádzajú merania priemerného času potrebného na prechod do ďalšej generácie evolučného algoritmu, tento prechod zahrňuje evauláciu, selekciu a obnovu populácie. Na grafe 6.5 znázorňujúcom závislosť času prechodu do ďalšej generácie EA (pri kalibrácii koeficientov logistickej a lineárnej regresie) na okolí je môžné pozorovať exponenciálny tvar krivky preloženej danými bodmi. Rovnice týchto exponenciál sú približne:

$$t_{Logistic}(x) \doteq 119,5618 \cdot 1,16449^x \tag{6.1}$$

$$t_{Linear}(x) \doteq 175,2122 \cdot 1,1388^x \tag{6.2}$$

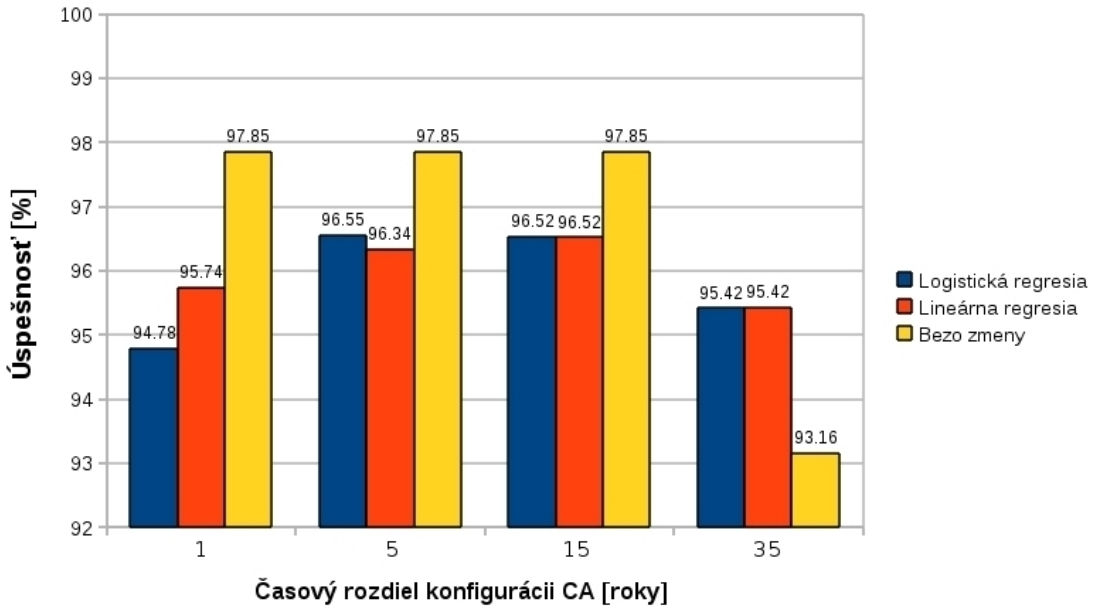
kde x je veľkosť okolia v jednej dimenzii. Základy exponenciálnych funkcií slúžiacich na výpočet času vykonania jednej generácie EA je približne rovnaký, pretože pri trénovaní nezávisí časová zložitosť výpočtu hodnoty, ktorá sa porovnáva s prahom, na okolí. Na okolí závisí len prepočet parametru, ktorý určuje počet zastavaných buniek v definovanom okolí. To znamená, že rozdiel časov trénovania prechodových funkcii na základe lineárnej a logistickej regresiu je konštantný bez ohľadu na veľkosť okolia. Tento rozdiel bude signifikantný hlavne pri malých okoliach, kde pomer medzi týmto rozdielom a celkovým časom je nezanedbatelný. V prípade väčších okolí sa však tento rozdiel bude relatívne k celkovému času zmenšovať. Z toho vyplýva, že časový rozdiel medzi časmi logistickej a lineárnej regresie bude stále konštantý bez ohľadu na definované parametre buniek.

Časový rozdiel 35 rokov						
Okolie	5	7	9	11	13	15
Fitness	409	389	387	384	384	381
Zastavané [%]	92,517	89,7321	95,2431	94,5902	96,1111	95,5996
Nezastavané [%]	95,4915	95,7426	95,2509	95,1962	95,0792	95,0973
Celkovo [%]	95,3421	**95,4157**	95,2506	95,1680	95,1243	95,1195
Čas [s]	7,10	7,83	8,81	10,03	11,50	13,22
Časový rozdiel 15 rokov						
Okolie	5	7	9	11	13	15
Fitness	1282	1316	1378	1322	1355	1319
Zastavané [%]	73,3361	72,4465	75,2258	75,5748	74,6134	71,6701
Nezastavané [%]	99,3341	99,2513	99,2007	99,1633	99,1349	99,1358
Celkovo [%]	96,2801	96,0907	96,4938	**96,5229**	96,3627	95,9013
Čas [s]	25,32	29,14	34,23	40,59	48,23	57,14
Časový rozdiel 5 rokov						
Okolie	5	7	9	11	13	15
Fitness	1302	1292	1271	1347	1294	1348
Zastavané [%]	75,4401	73,5627	64,5742	68,0296	69,8559	67,5969
Nezastavané [%]	99,2444	99,2202	99,2389	99,1789	99,1710	99,1617
Celkovo [%]	**96,5521**	96,251	94,6533	95,2943	95,6148	95,2069
Čas [s]	72,50	90,48	114,45	144,42	180,37	222,33
Časový rozdiel 1 rok						
Okolie	5	7	9	11	13	15
Fitness	1375	1458	1539	1342	1370	–
Zastavané [%]	64,7985	65,3688	62,8551	51,9976	54,4949	–
Nezastavané [%]	99,2946	99,1686	99,1747	99,1983	99,2452	–
Celkovo [%]	94,7213	**94,7844**	94,2696	91,453	92,2106	–
Čas [s]	329,71	434,30	573,75	748,07	957,25	–
Prechodová funkcia zachovávajúca počiatočnú konfiguráciu						
	časový rozdiel 35 rokov			časový rozdiel 1, 5, 15 rokov		
Fitness	793			2202		
Zastavané [%]	45, 9376			89, 2207		
Nezastavané [%]	99, 9722			99, 1763		
Celkovo [%]	93, 1575			97, 8487		

Tabuľka 6.5: Výsledky experimentov, ktoré vytvárali prechodové funkcie na základe logistickej regresie

Časový rozdiel 35 rokov						
Okolie	5	7	9	11	13	15
Fitness	411	392	380	411	378	375
Zastavané [%]	85,8407	94,7712	88,2409	97,1583	92,3664	93,641
Nezastavané [%]	96,036	95,1001	95,3392	94,3787	94,9835	95,3051
Celkovo [%]	**95,4206**	95,0855	94,9786	94,4784	94,8669	95,2263
Čas [s]	5,17	5,89	6,86	8,06	9,51	11,2
Časový rozdiel 15 rokov						
Okolie	5	7	9	11	13	15
Fitness	1273	1360	1340	1316	1295	1303
Zastavané [%]	70,407	74,2411	75,4098	75,1405	74,8705	70,6911
Nezastavané [%]	99,3743	99,2645	99,1956	99,1411	99,1191	99,1396
Celkovo [%]	95,8139	96,3821	**96,5181**	96,4452	96,3918	95,7411
Čas [s]	18,60	23,09	29,07	36,55	45,53	56,00
Časový rozdiel 5 rokov						
Okolie	5	7	9	11	13	15
Fitness	1275	1301	1263	1303	1265	1294
Zastavané [%]	74,0616	73,2441	69,167	70,3465	68,7056	65,4437
Nezastavané [%]	99,2371	99,2143	99,2082	99,1773	99,1472	99,1797
Celkovo [%]	**96,3384**	96,1976	95,5128	95,7022	95,4011	94,8038
Čas [s]	50,8	70,31	96,33	128,85	167,87	213,40
Časový rozdiel 1 rok						
Okolie	5	7	9	11	13	15
Fitness	1446	1373	1480	1529	–	–
Zastavané [%]	60,2851	64,3881	70,6456	57,2775	–	–
Nezastavané [%]	99,3256	99,1998	99,1450	99,2463	–	–
Celkovo [%]	93,7403	94,5998	**95,7362**	92,973	–	–
Čas [s]	244,50	349,42	489,30	664,16	–	–
Prechodová funkcia zachovávajúca počiatočnú konfiguráciu						
	časový rozdiel 35 rokov			časový rozdiel 1, 5, 15 rok		
Fitness	793			2202		
Zastavané [%]	45, 9376			89, 2207		
Nezastavané [%]	99, 9722			99, 1763		
Celkovo [%]	93, 1575			97, 8487		

Tabuľka 6.6: Výsledky experimentov, ktoré vytvárali prechodové funkcie na základe lineárnej regresie



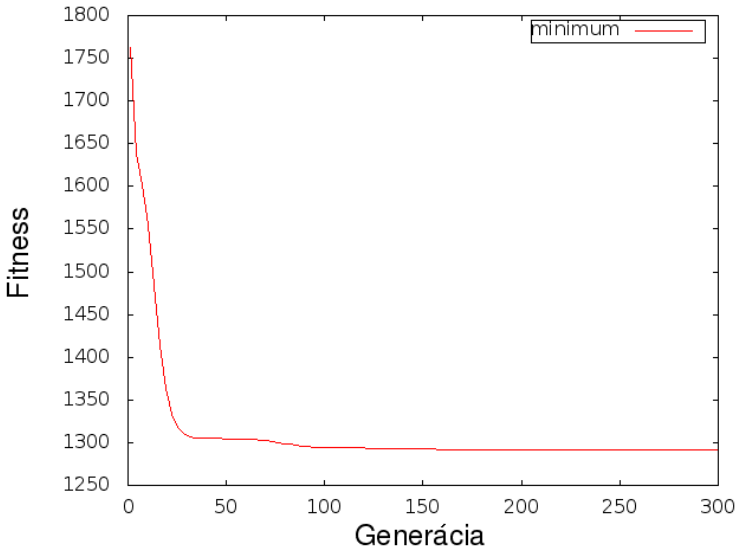
Obrázok 6.2: Porovnanie najlepších úspešnosti prechodových funkcí založených na lineárnej regresii, logistickej regresii a prechodovej funkcii, ktorá zachováva počiatočnú konfiguráciu bezo zmeny, v závislosti na časovom rozdiele konfigurácii CA

Koeficient logistickej regresie		**Hodnota**
α_{Urban}	Konštanta	0,3689
$\beta_{Urban,1}$	Váha okolia	1
$\beta_{Urban,2}$	Váha vzdialenosti od centra	0,3041
$\beta_{Urban,3}$	Váha vzdialenosti od lesov	0,1427
$\beta_{Urban,4}$	Váha nadmorskej výšky	-0,0477
γ_{Urban}	Prah pre stav zastavanosti	0,6167
$\gamma_{Nonurban}$	Prah pre stav nezastavanosti	0,4196

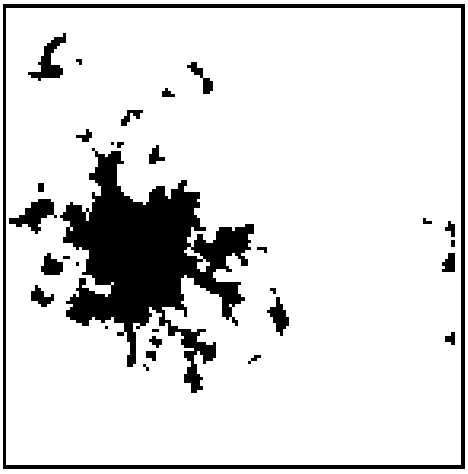
Tabuľka 6.7: Koeficienty logistickej regresie pre 5-okolie pri časovom rozdiele stavov CA 5 rokov

Koeficient lineárnej regresie		Hodnota
α_{Urban}	Konštanta	0,5824
$\beta_{Urban,1}$	Váha okolia	1
$\beta_{Urban,2}$	Váha vzdialenosti od centra	0,421
$\beta_{Urban,3}$	Váha vzdialenosti od lesov	-0,0085
$\beta_{Urban,4}$	Váha nadmorskej výšky	0,5250
γ_{Urban}	Prah pre stav zastavanosti	1,3658
$\alpha_{Nonurban}$	Konštanta	0,8592
$\beta_{Nonurban,1}$	Váha okolia	-0,5304
$\beta_{Nonurban,2}$	Váha vzdialenosti od centra	1
$\beta_{Nonurban,3}$	Váha vzdialenosti od lesov	-1
$\beta_{Nonurban,4}$	Váha nadmorskej výšky	0,1912
$\gamma_{Nonurban}$	Prah pre stav nezastavanosti	2,8836

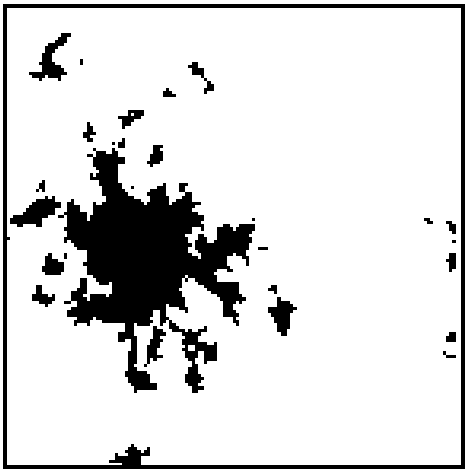
Tabuľka 6.8: Koeficienty lineárnej regresie pre 9-okolie pri časovom rozdiele stavov CA 15 rokov



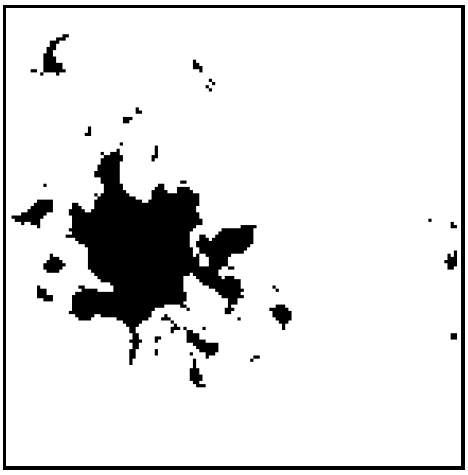
Obrázok 6.3: Konvergencia evolučného algoritmu



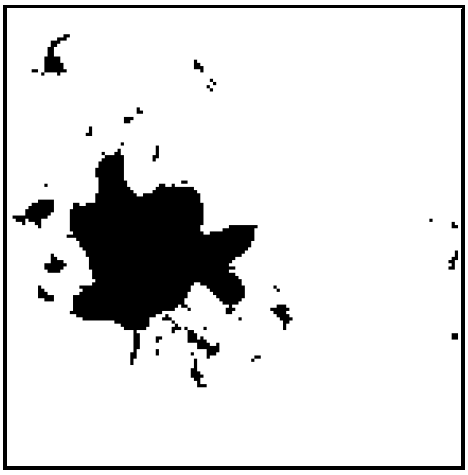
(a) trénovací stav roku 1990 = počiatočná konfigurácia



(b) testovací stav roku 2005

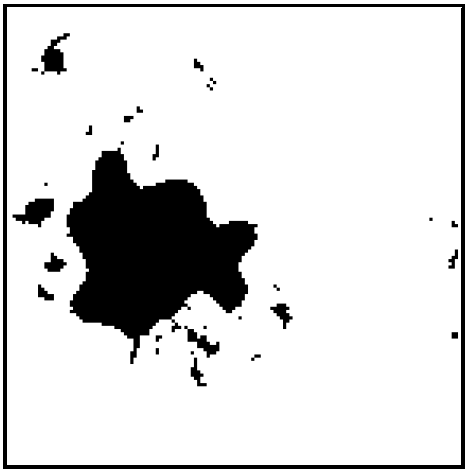


(c) simulovaný stav roku 2005 pomocou logistickej regresie



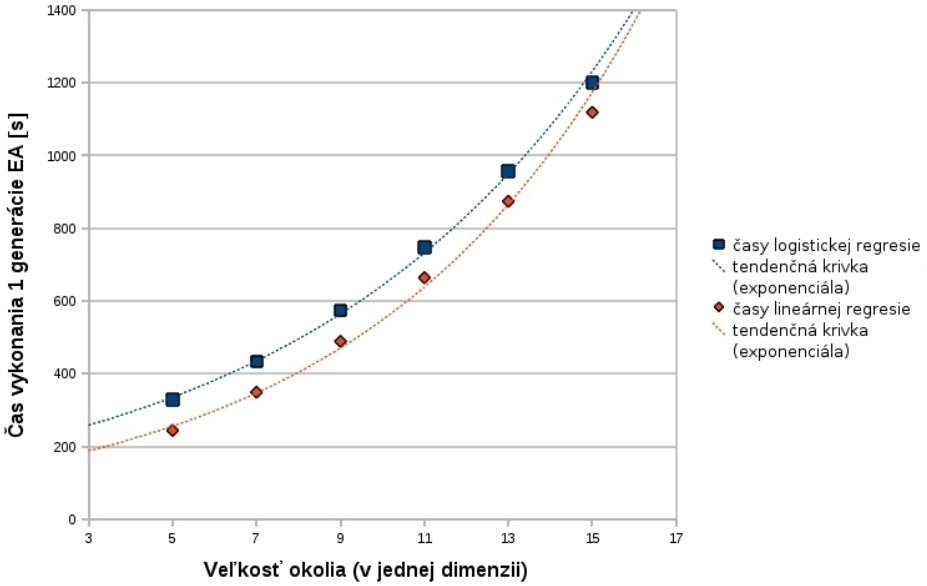
(d) simulovaný stav roku 2005 pomocou lineárnej regresie

(e) simulovaný stav roku 2020 pomocou logistickej regresie



(f) simulovaný stav roku 2020 pomocou lineárnej regresie

Obrázok 6.4: Simulované stavy urbanizácie Brna pre 5-okolie a časový rozdiel konfigurácii CA 5 rokov (logistická regresia) resp. pre 9-okolie a časový rozdiel konfigurácii CA 15 rokov (lineárna regresia)



Obrázok 6.5: Čas vykonania jednej generácie evolučného algoritmu hľadajúceho optimálne koeficienty logistickej resp. lineárnej regresie v závislosti na veľkost okolia

Kapitola 7

Záver

Počítačové simulátory sú v súčasnosti vhodným nástrojom na simuláciu reálnych systémov. Častokrát je však vzťahov v modelovanom systéme priveľa a často sú tieto vzťahy nelineárne. V takýchto prípadoch nie sme schopný rigoróznym spôsobom definovať model systému a siahame k modelom využívajúcich rôzne optimalizačné techniky. Jedným z nich je model založený na celulárnych automatoch spolupracujúci s evolučnými algoritmami. Tento model je vhodný na simuláciu hlavne priestorových dynamických javov vykazujúcich komplexné chovanie. Cieľom tejto práce bolo vytvoriť simulátor, ktorý by slúžil na modelovanie takýchto javov. Na popis a analýzu vzťahov využíva princípy lineárnej a logistickej regresie. Evolučný algoritmus hľadá vhodné váhy parametrov alebo vlastností buniek celulárneho automatu, ktoré predstavujú koeficienty lineárnej a logistickej regresie.
Možné vylepšenia simulátoru:

- pridať alternatívny spôsob hľadania vzťahov z dát – polynomiálnu regresiu, diskriminačnú analýzu, Bayesovské siete, neurónové siete alebo využiť tzv. kernel – based CA
- použitie adaptívnych evolučných algoritmov, ktoré dokážu za chodu evolučného algoritmu meniť parametre evolučných operátorov
- zefektívnenie výpočtu simulátoru, napríklad využitím paralelného prístupu
- implementácia tzv. segmentového celulárneho automatu, ktorý by bol rozdelený na segmenty a prechodová funkcia by sa hľadala pre každý segment zvášť

Pomocou simulátoru sme sa pokúsili predikovať rast urbanizácie mesta Brna na základe dostupných máp a parametrov, ktorých hodnoty sú definované pre všetky pixely máp. Na tento problém sme aplikovali obe techniky regresnej analýzy – lineárnu aj logistickú regresiu. Výsledky boli porovnatelné. Ako výhodná sa v tomto prípade zdá byť klasická lineárna regresia pracujúca s prahom, pretože modelované vzťahy sú skôr lineárneho charakteru. Pri trénovaní sme menili veľkosť okolia CA (5,7,9,11,13,15) a časový odstup medzi konfiguráciami CA (1,5,15,35 rokov). Lepšie výsledky dávali skôr menšie okolia ako okolia rozsiahle. Výsledky natrénovaných prechodových funkcií neboli najlepšie, čo bolo spôsobené hlavne týmito faktormi:

- nedostatok parametrov buniek CA
- neboli implementované žiadne obmedzujúce podmienky ako je napríklad použitie tzv. Tietenbergovho populačného modelu

- neboli zohľadnené socio-politické procesy
- časový odstup trénovacích dát bol pomerne veľký
- vhodné by bolo rozdeliť oblasť na podoblasti a nepredikovať rozľahlé oblasti ako celok, pretože tak nie sme schopný zachytiť rôzne lokálne vplyvy

Popri experimentoch sme merali aj časy života jednej generácie populácie vrátane evaluácie, selekcie a obnovy populácie. Časový rozdiel medzi trénovaním koeficientov lineárnej a logistickej regresie je konštantný a nezávislý na veľkosti okolia resp. nezávislý na parametroch buniek CA. Keďže je tento časový rozdiel konštantný, je znatelný hlavne vtedy, ak ostatné operácie, ako napríklad prepočítavanie rôznych parametrov buniek, nebudú veľmi časovo náročné. V opačnom prípade sa tento rozdiel stiera a môžeme použiť logistickú regresiu s porovnatelnou časovou zložitosťou použitia lineárnej regresie.

Naša predikcia rastu urbanizácie Brna zrejme nebude mať praktické uplatnenie, ale môže byť základom pre komplexnejšiu predikciu.

Literatura

[1] Agresti, A.: *An introduction to categorical data analysis*. Wiley series in probability and statistics, Hoboken, NJ: Wiley-Interscience, druhé vydání, 2007, ISBN 978-0-471-22618-5.

[2] Al-Ahmadi, K.; Heppenstall, A.; Hogg, J.; aj.: A Fuzzy Cellular Automata Urban Growth Model (FCAUGM) for the City of Riyadh, Saudi Arabia. Part 1: Model Structure and Validation. *Applied Spatial Analysis and Policy*, ročník 2, 2009: s. 65–83, ISSN 1874-463X.

[3] Berry, W.; Feldman, S.: *Multiple regression in practice*. Sage Publications, 1985, ISBN 0803920547.

[4] Brannick, M. T.: Logistic Regression. [online], [cit. 2011-04-10].
URL http://luna.cas.usf.edu/~mbrannic/files/regression/Logistic.html

[5] Cecchini, A.; Rinaldi, E.: The multi - cellular automaton: a tool to build more sophisticated models. A theoretical foundation and a practical implementation. 1999.

[6] Chopard, B.: Cellular Automata Modeling of Physical Systems. In *Encyclopedia of Complexity and Systems Science*, editace R. A. Meyers, Springer, 2009, ISBN 978-0-387-75888-6.

[7] Das, R.; Crutchfield, J. P.; Mitchell, M.; aj.: Evolving Globally Synchronized Cellular Automata. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, Morgan Kaufmann, 1995, s. 336–343.
URL http://cse.ucdavis.edu/~evca/Papers/EGSCA.pdf

[8] Demek, J.; Havlíček, M.; Mackovčin, P.; aj.: Brno and its surroundings: A landscape-ecological study. In *Ekologie krajiny (Journal of Landscape Ekology)*, 2007, ISBN 978-80-86386-97-3.
URL http://www.iale.cz/downloads/JLE_0/032%20-%20053.pdf

[9] Eiben, A. E.; Smith, J. E.: *Introduction to Evolutionary Computing*. SpringerVerlag, 2003, ISBN 3540401849.

[10] Fuqiang, D.: Mining Dynamic Transition Rules of Cellular Automata in Urban Population Simulation. In *Proceedings of the 2010 Second International Conference on Computer Modeling and Simulation - Volume 02*, ICCMS '10, Washington, DC, USA: IEEE Computer Society, 2010, ISBN 978-0-7695-3941-6, s. 471–474.

[11] Greene, B.: *The elegant universe : superstrings, hidden dimensions, and the quest for the ultimate theory*. Vintage Books, New York, první vydání, 2000, ISBN 0375708111, 448 s.

[12] Guan, Q.; Wang, L.: An Artificial-Neural-Network-Based Constrained CA Model for Simulating Urban Growth and Its Application. 2005.
URL http://www.geog.ucsb.edu/~guan/papers/Guan_AutoCarto2005.pdf

[13] Hawking, S.: *A Brief History of Time*. New York: Bantam Dell Publishing Group, 10 vydání, 1996, ISBN 9780553109535.

[14] Husáková, M.: Celulární automaty. [online],[cit. 2011-02-10].
URL http://lide.uhk.cz/fim/ucitel/fshusam2/lekarnicky/zt3/zt3_dokumenty/CelularniAutomaty.pdf

[15] Hynek, J.: *Genetické algoritmy a genetické programovanie*. Praha 7: Grada Publishing a.s., 2008, ISBN 978-80-247-2695-3.

[16] Kalátová, E.; Dobiáš, J.: Evoluční algoritmy. [online],[cit. 2011-03-11].
URL http://www.kiv.zcu.cz/studies/predmety/uir/gen_alg2/E_alg.htm

[17] Kapoun, J.: Nový druh vědy si dobře rozumí s byznysem. 2005, [online],[cit. 2011-02-07].
URL http://scienceworld.cz/technologie/novy-druh-vedy-si-dobe-rozumi-s-byznysem-1715

[18] Kleinbaum, D.; Klein, M.; Pryor, E.: *Logistic Regression: A Self-Learning Text*. Statistics for Biology and Health, Springer, 2010, ISBN 9781441917416.
URL http://books.google.cz/books?id=J7E0JQweHkoC

[19] Kocabas, V.; Dragicevic, S.: Coupling Bayesian Networks with GIS-Based Cellular Automata for Modeling Land Use Change. In *Geographic, Information Science*, editace M. Raubal; H. Miller; A. Frank; M. Goodchild, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2006, s. 217–233.

[20] Koza, J. R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*. The MIT Press, první vydání, December 1992, ISBN 0262111705.

[21] Li, X.; Yang, Q.; Liu, X.: Genetic algorithms for determining the parameters of cellular automata in urban simulation. *Science in China Series D: Earth Sciences*, ročník 50, 2007: s. 1857–1866, ISSN 1006-9313.

[22] Mahajan, Y.; Venkatachalam, P.: Neural Network Based Cellular Automata Model for Dynamic Spatial Modeling in GIS. In *Proceedings of the International Conference on Computational Science and Its Applications: Part I*, ICCSA '09, Berlin, Heidelberg: Springer-Verlag, 2009, ISBN 978-3-642-02453-5, s. 341–352.

[23] Menard, S.: *Applied logistic regression analysis*. Thousand Oaks, CA: Sage Publications, druhé vydání, 2002, ISBN 0-7619-2208-3.

[24] Mitchell, M.; Crutchfield, J. P.; Das, R.; aj.: Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work. 1996.
URL http://cse.ucdavis.edu/~evca/Papers/evca-review.pdf

[25] Park, T.; Ryu, K. R.: A Dual-Population Genetic Algorithm for Adaptive Diversity Control. *Evolutionary Computation, IEEE Transactions on*, ročník 14, č. 6, december 2010: s. 865–884, ISSN 1089-778X.

[26] Roelofs, G.: PNG Documentation. 2010, [online],[cit. 2011-05-01].
URL http://www.libpng.org/pub/png/pngdocs.html

[27] Schwarz, J.; Růžička, R.; Strnadel, J.: Mikroprocesorové a vestavěné systémy: Studijná opora. 2006, [online],[cit. 2011-01-19].

[28] Schwarz, J.; Sekanina, L.: Aplikované evolučné algoritmy: Studijná opora. 2006, [online],[cit. 2011-02-20].

[29] Sipper, M.: *Evolution of Parallel Cellular Machines: The Cellular Programming Approach*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2001, ISBN 3540626131.

[30] Soares-Filho, B. S.; Cerqueira, G. C.; Pennachin, C. L.: A stochastic cellular automata model designed to simulate the landscape dynamics in an Amazonian colonization frontier. *Ecological Modelling*, ročník 154, č. 3, 2002: s. 217 – 235, ISSN 0304-3800.

[31] Tyler, T.: The Margolus neighbourhood. [online],[cit. 2011-03-01].
URL http://cell-auto.com/neighbourhood/margolus/

[32] Wall, M.: GAlib: A C++ library of genetic algorithm components. *Mechanical Engineering Department, Massachusetts Institute of Technology*, 1996, [online],[cit. 2011-05-05].

[33] Wikipedie: Geografický informační systém — Wikipedie, otevřená encyklopedie. [online],[cit. 2011-04-14].
URL http://cs.wikipedia.org/wiki/Geografický_informační_systém

[34] Wikipedie: Regresní analýza — Wikipedie, otevřená encyklopedie. [online],[cit. 2011-03-15].
URL http://cs.wikipedia.org/wiki/Regresní_analýza

[35] Wolfram, S.: *A New Kind of Science*. Wolfram Media, January 2002, ISBN 1-57955-008-8, 1197 s.

[36] Yeh, A.; Xia, L.: Integration of neural networks and cellular automata for urban planning. *Geo-Spatial Information Science*, ročník 7, 2004: s. 6–13, ISSN 1009-5020.

[37] Zhang, F.; Pu, L.; Ding, L.; aj.: Calibration of cellular automata model with adaptive genetic algorithm for the simulation of urban land-use. In *Geoinformatics, 2010 18th International Conference on*, june 2010, s. 1 –6.

[38] Šíma, J.; Neruda, R.: *Teoretické otázky neuronových sítí*. Praha: Matfyzpress, 1996, ISBN 80-85863-18-9.
URL http://www2.cs.cas.cz/~sima/kniha.html

Příloha A

DTD vstupných a výstupných dát simulátora

XML DTD pre vstupné/výstupné dáta:

```
<!ELEMENT ca (info, cells)>
<!ATTLIST ca d1 CDATA "">
<!ATTLIST ca d2 CDATA "">
<!ATTLIST ca d3 CDATA "">
<!ELEMENT info (name?, description?)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT cells (cell*)>
<!ELEMENT cell (params?)>
<!ATTLIST cell state CDATA "0">
<!ATTLIST cell d1 CDATA "0">
<!ATTLIST cell d2 CDATA "1">
<!ATTLIST cell d3 CDATA "2">
<!ELEMENT param (#PCDATA)>
<!ATTLIST param name CDATA "">
```

XML DTD pre definíciu parametrov:

```
<!ELEMENT caparams (param*)>
<!ELEMENT param (name,max)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT max (#PCDATA)>
```